Chapter 2

Polynomial Chaos Expansion for Probabilistic Uncertainty Propagation

Shuxing Yang, Fenfen Xiong and Fenggang Wang

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.68484

Abstract

Uncertainty propagation (UP) methods are of great importance to design optimization under uncertainty. As a well-known and rigorous probabilistic UP approach, the polynomial chaos expansion (PCE) technique has been widely studied and applied. However, there is a lack of comprehensive overviews and studies of the latest advances of the PCE methods, and there is still a large gap between the academic research and engineering application for PCE due to its high computational cost. In this chapter, latest advances of the PCE theory and method are elaborated, in which the newly developed data-driven PCE method that does not depend on the complete information of input probabilistic distribution as the common PCE approaches is introduced and improved. Meanwhile, the least angle regression technique and the trust region scenario are, respectively, extended to reduce the computational cost of data-driven PCE to accommodate it to practical engineering design applications. In addition, comprehensive comparisons are made to explore the relative merits of the most commonly used PCE approaches in the literature to help designers to choose more suitable PCE techniques in probabilistic design optimization.

Keywords: uncertainty propagation, probabilistic design, polynomial chaos expansion, data-driven, sparse, trust region

1. Introduction

Uncertainties are ubiquitous in engineering problems, which can roughly be categorized as aleatory and epistemic uncertainty [1, 2]. The former represents natural or physical randomness that cannot be controlled or reduced by designers or experimentalists, while the latter

refers to reducible uncertainty resulting from a lack of data or knowledge. In systems design, all sources of uncertainties need to be propagated to assess the uncertainty of system quantities of interest, i.e., uncertainty propagation (UP). As is well known, UP is of great importance to design under uncertainty, which greatly determines the efficiency of the design. Since generally sufficient data are available for aleatory uncertainties, probabilistic methods are commonly employed for computing response distribution statistics based on the probability distribution specifications of input [3, 4]. Conversely, for epistemic uncertainties, data are generally sparse, making the use of probability distribution assertions questionable and typically leading to nonprobabilistic approaches, such as the fuzzy, evidence, and interval theories [5–7]. This chapter mainly focuses on propagating the aleatory uncertainties to assess the uncertainty of system quantities of interest using probabilistic methods, which is shown in **Figure 1**.

A wide variety of probabilistic UP approaches for the analysis of aleatory uncertainties have been developed [8], among which the polynomial chaos expansion (PCE) technique is a rigorous approach due to its strong mathematical basis and ability to produce functional representations of stochastic quantities. With PCE, the function with random inputs can be represented as a stochastic metamodel, based on which lower-order statistical moments as well as reliability of the function output can be derived efficiently to facilitate the implementation of design optimization under uncertainty scenarios such as robust design [9] and reliability-based design [10]. The original PCE method is an intrusive approach in the sense that it requires extensive modifications in existing deterministic codes of the analysis model, which is generally limited to research where the specialist has full control of all model equations as well as detailed knowledge of the software. Alternatively, nonintrusive approaches have been developed without modifying the original analysis model, gaining increasing attention, thus is the focus of this chapter. As a well-known PCE approach, the generalized PCE (gPCE) method based on the Askey scheme [11, 12] has been widely applied to UP for its higher accuracy and better convergence [13, 14] compared to the classic Wiener PCE [15]. Generally, the random input does not necessarily follow the five types of probabilistic distributions (i.e., normal, uniform, exponential, beta, and gamma) in the Askey scheme. In this case, the transformation should be made to transfer each random input variable to one of the five distributions. It would induce substantially lower convergence rate, which makes the

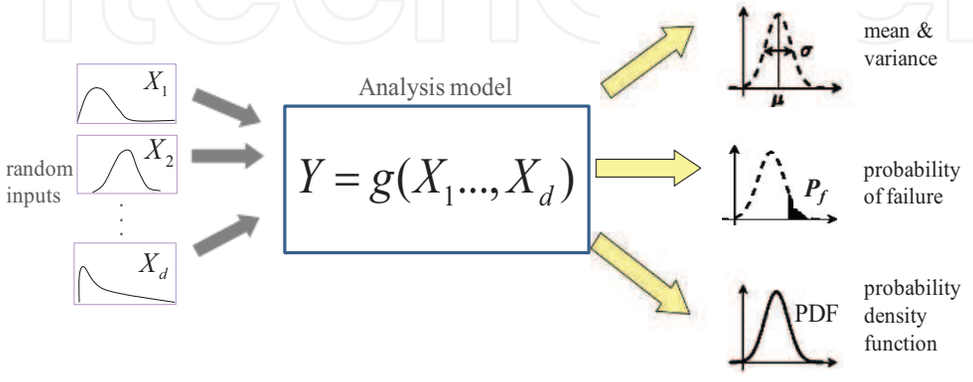

Figure 1. Illustration of uncertainty propagation.

nonoptimal application of Askey polynomial chaos computationally inefficient [8]. Therefore, the Gram-Schmidt PCE (GS-PCE) [16] and multielement PCE (ME-PCE) [17] methods have been developed to accommodate arbitrary distributions through constructing their own orthogonal polynomials rather than referring to the Askey scheme.

All the PCE methods discussed above are constructed based on the assumption that the exact knowledge of the involved joint multivariate probability density function (PDF) of all random input variables exists. Generally, by assumption of independence of the random variables, the joint PDF is factorized into univariate PDFs of each random variable in introducing PCE in the literature. However, the random input could exist as some raw data with a complicated cumulative histogram, such as bi-modal or multi-modal type, for which it is often difficult to obtain the analytical expression of its PDF accurately. Under these scenarios, all the above PCE approaches become ineffective since they all have to assume the PDFs to be complete. To address this issue, the data-driven PCE (DD-PCE) method has been proposed [18], in which its accuracy and convergence with diverse statistical distributions and raw data are tested and well demonstrated. With this PCE method, the one-dimensional orthogonal polynomial basis is constructed directly based on a set of data of the random input variables by matching certain order of their statistic moments, rather than the complete distributions as in the existing PCE methods, including gPCE, GS-PCE, and ME-PCE.

At present, great research achievements about PCE have been made in the literature, which have also been applied to practical engineering problems to save the computational cost in UP. However, there is still a large gap between the academic study and engineering application for the PCE theory due to the following reasons: (1) the complete information of input PDF often is not known in engineering, which cannot be solved by most PCE methods presented in the literature; (2) the computational cost of existing PCE approaches is still very high, which cannot be afforded in practical problems, especially when applied to design optimization; and (3) there is a lack of comprehensive exploration of the relative merits of all the PCE approaches to help designers to choose more suitable PCE techniques in design under uncertainty.

2. Data-driven polynomial chaos expansion method

Most PCE methods presented in the literature are constructed based on the assumption that the exact knowledge of the involved PDF of each random input variable exists. However, the PDF of a random parameter could exist as some raw data or numerically as a complicated cumulative histogram, such as bimodal or multimodal type, which is often difficult to obtain the analytical expression of its PDF accurately. To address this issue, the data-driven PCE method (DD-PCE for short in this chapter) has been proposed. DD-PCE follows the similar general procedure as that of the well-known gPCE method. For gPCE, the one-dimensional orthogonal polynomial basis simply comes from the Askey scheme in **Table 1** and is a function of standard random variables. While for DD-PCE, the one-dimensional orthogonal polynomial basis is constructed directly based on the data of random input by matching certain order of statistic moments of the random inputs and is a function of the original random variables.

Distribution types	PDFs	Polynomials	Weights	Intervals
Normal	$\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$	Hermite $H_n(x)$	$e^{-x^2/2}$	$[-\infty, +\infty]$
Uniform	1/2	Legendre $P_n(x)$	1	$[-1, 1]$
Beta	$\frac{(1-x)^\alpha(1+x)^\beta}{2^{\alpha+\beta+1}B(\alpha+1,\beta+1)}$	Jacobi $P_n^{(\alpha,\beta)}(x)$	$(1-x)^\alpha(1+x)^\beta$	$[-1, 1]$
Exponential	e^{-x}	Laguerre $L_n(x)$	e^{-x}	$[0, +\infty]$
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	General Laguerre $L_n^{(\alpha,\beta)}$	$x^\alpha e^{-x}$	$[0, +\infty]$

Table 1. Random variable types and the corresponding orthogonal polynomials.

2.1. Procedure of data-driven PCE method

Step 1. Represent the output y as a PCE model of order p.

$$y \approx \sum_{i=0}^{P} b_i \Phi_i(\boldsymbol{X}) = \sum_{i=0}^{P} b_i \prod_{j=1}^{d} P_j^{(\alpha_j^i)}(X_j) \tag{1}$$

where P+1 $(1 + P = (d + p)!/(d!p!))$ is the number of PCE coefficients b_i that is the same as gPCE; $\Phi_i(\boldsymbol{X})$ is the d-dimensional orthogonal polynomial produced by the full tensor product of one-dimensional orthogonal polynomials $P_j^{(\alpha_j^i)}(X_j)$; and α_j^i represents the order of $P_j^{(\alpha_j^i)}(X_j)$ and clearly satisfies $0 \le \sum_{j=1}^{d} \alpha_j^i \le p$.

$P_j^{(\alpha_j^i)}(X_j)$ corresponding to the jth dimensional random input variable x_j in Eq. (1) is defined as below, in which the index α_j^i is replaced by k_j for simplicity below:

$$P_j^{(k_j)}(X_j) = \sum_{s=0}^{k_j} p_{s,j}^{(k_j)} * (X_j)^s, \; j = 1, 2, \ldots, d \tag{2}$$

where $p_{s,j}^{(k_j)}$ is the unknown polynomial coefficient to be solved.

Step 2. Solve the unknown polynomial coefficient $p_{s,j}^{(k_j)}$ to construct the one-dimensional orthogonal polynomial basis.

Since the construction of $P_j^{(\alpha_j^i)}(X_j)$ on each dimension is the same, the subscript j denoting the dimension number is omitted thereafter for simplicity. Based on the property of orthogonality, one clearly has

$$\int_{x \in \Omega} P^{(k)}(X) P^{(l)}(X) d\Gamma(X) = \delta_{kl}, \forall k, l = 0, 1, \ldots, p \tag{3}$$

where δ_{kl} is the Kronecker delta, Ω is the original stochastic span, and $\Gamma(X)$ represents the cumulative distribution function of the random variable X.

It is assumed that all the coefficients $p_s^{(k)}$ in Eq. (2) are not equal to 0, and then $P^{(0)} = p_0^{(0)}$. For simplicity, the coefficient of the highest degree term in each $P^{(k)}$ is set as $p_k^{(k)} = 1, \forall k$. According to Eq. (3), one has

$$\int_{x\in\Omega} p_0^{(0)} \left[\sum_{s=0}^{k} p_s^{(k)} X^s\right] d\Gamma(X) = 0 \tag{4}$$

In the same way as above, one has

$$\begin{aligned} &\int_{x\in\Omega} \left[\sum_{s=0}^{1} p_s^{(1)} X^s\right] \left[\sum_{s=0}^{k} p_s^{(k)} X^s\right] d\Gamma(X) = 0 \\ &\qquad\vdots \qquad\qquad\qquad\vdots \\ &\int_{x\in\Omega} \left[\sum_{s=0}^{k-1} p_s^{(k-1)} X^s\right] \left[\sum_{s=0}^{k} p_s^{(k)} X^s\right] d\Gamma(X) = 0 \end{aligned} \tag{5}$$

There are totally k equations in Eqs. (4) and (5). Through substituting Eq. (4) into the first equation in Eq. (5), and then substituting Eq. (4) and the first equation in Eq. (5) to the second equation in Eq. (5), and so on, one set of new equations can be derived:

$$\begin{aligned} &\int_{x\in\Omega} \sum_{s=0}^{k} p_s^{(k)} X^s d\Gamma(X) = 0 \\ &\int_{x\in\Omega} \sum_{s=0}^{k} p_s^{(k)} X^{s+1} d\Gamma(X) = 0 \\ &\qquad\vdots \\ &\int_{x\in\Omega} \sum_{s=0}^{k} p_s^{(k)} X^{s+k-1} d\Gamma(X) = 0 \end{aligned} \tag{6}$$

It is observed that $\int_{\xi\in\Omega} X^k d\Gamma(X)$ is actually the kth order statistic moment of x, i.e., $\int_{x\in\Omega} X^k d\Gamma(X) = \mu_k$. Therefore, Eq. (6) can be rewritten as

$$\begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_k \\ \mu_1 & \mu_2 & \cdots & \mu_{k+1} \\ \vdots & \vdots & \vdots & \vdots \\ \mu_{k-1} & \mu_k & \cdots & \mu_{2k-1} \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} p_0^{(k)} \\ p_1^{(k)} \\ \vdots \\ p_{k-1}^{(k)} \\ p_k^{(k)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{7}$$

where $\mu_i (i = 0, 1, \ldots, 2k-1)$ is the ith order statistic moment of x, which can be easily calculated from the given input data statistically or the PDFs of random inputs by integral. Of course, when the number of given data is not large enough, errors would be induced in the moment calculation.

Clearly, to obtain a k-order one-dimensional orthogonal polynomial basis, 0 to $(2k-1)$-order statistic moments of x should be matched, which can be calculated based on the PDF or statistically

based on the data set. Of course, when the number of data is not large enough, errors would be induced in the moment calculation. The polynomial coefficients for the one-dimensional orthogonal polynomial basis can be obtained by solving Eq. (7) with the Cramer's Rule.

Step 3. Calculate the PCE coefficients b_i by the least square regression technique.

Step 4. Once the PCE coefficients are obtained, a stochastic metamodel (i.e., PCE model) that is much cheaper than the original model is provided. Evaluate on the PCE model by Monte Carlo simulation (MCS) to obtain the probabilistic characteristics of y. Since the PCE model is cheap, a large amount of sample points can be used. For the statistic moments, the analytical expressions can also be conveniently derived based on the PCE coefficients:

$$\begin{cases} E[y] = E\left[\sum_{i=0}^{P} b_i \psi_i(\boldsymbol{X})\right] = b_0 \\ \sigma^2[y] = E[y^2] - E^2[y] = \sum_{i=0}^{P} b_i^2 E[\psi_i^2(\boldsymbol{X})] - E^2[y] \\ Skew[y] = E\left[\left(\frac{y-E[y]}{\sigma[y]}\right)^3\right] = \frac{E[y^3] - 3E[y]\sigma^2[y] - E^3[y]}{\sigma^3[y]} \\ Kur[y] = \frac{E[(y-E[y])^4]}{\sigma^4[y]} = \frac{E[y^4] - 4E[y]E[y^3] + 6E^2[y]\sigma^2[y] + 3E^4[y]}{\sigma^4[y]} \end{cases} \tag{8}$$

2.2. Extension of Galerkin projection to DD-PCE

In the existing work about DD-PCE, only the regression method is employed to calculate the PCE coefficients. To the experience of the authors, the matrix during regression may become ill-conditioned during regression for higher-dimensional problems since the sample points required for regression that is often set as two times of the number of PCE coefficients $P + 1$ [19] is increased greatly causing a large-scale matrix during regression. To solve higher-dimensional problems, the Galerkin projection method in conjunction with the sparse grid technique has been widely used in gPCE due to its high accuracy, robustness, and convergence [20], which has also been observed and demonstrated during our earlier studies on PCE in recent years. In this section, the Galerkin projection method for PCE coefficients calculation is extended to the DD-PCE approach to address higher-dimensional UP problems. **Figure 2** shows the general procedure of the improved DD-PCE method.

With the projection method, the Galerkin projection is conducted on each side of Eq. (1):

$$\langle y\Phi_j(\boldsymbol{X})\rangle = \left\langle \sum_{i=0}^{P} b_i \Phi_i(\boldsymbol{X})\Phi_j(\boldsymbol{X}) \right\rangle, (j = 0, 1, \cdots, P) \tag{9}$$

where $\langle \bullet \rangle$ represents the operation of inner product as below

$$\langle g, f\rangle = \int g f dH(\boldsymbol{X}) \tag{10}$$

where H($\boldsymbol{X}$) is the joint cumulative distribution function of random input variables $\boldsymbol{X}$.

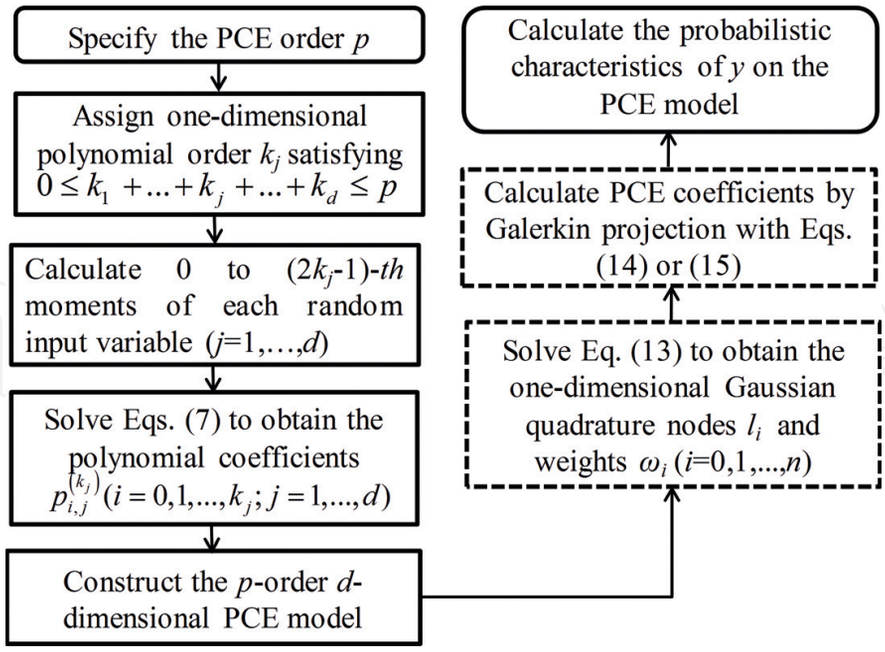

Figure 2. Procedure of the improved DD-PCE.

Based on the orthogonality property of orthogonal polynomials, the PCE coefficient can be calculated as

$$b_i = E[y\Phi_i(\boldsymbol{X})]/E[\Phi_i(\boldsymbol{X})\Phi_i(\boldsymbol{X})], (i = 0, 1, \cdots, P) \tag{11}$$

Similar to gPCE, the key point is the computation of the numerator in Eq. (11), which can be expressed as

$$E[y\Phi_i(\boldsymbol{X})] = \int_{\xi \in \Omega} y\Phi_i(\boldsymbol{X})dH(\boldsymbol{X}) \tag{12}$$

The Gaussian quadrature technique, such as full factorial numerical integration (FFNI) and spare grid numerical integration, has been widely used to calculate the numerator in the existing gPCE approaches, with which the one-dimensional Gaussian quadrature nodes and weighs are directly derived by multiplying some scaling factors on the nodes and weights from the existing Gaussian quadrature formulae and then the tensor product is employed to obtain the multidimensional nodes. For some common type of probability distributions, for example, normal, uniform, and exponential distributions, their PDFs have the similar formulations as the weighting functions of the Gaussian-Hermite, Gaussian-Legendre, and Gaussian-Laguerre quadrature formula. Therefore, l_i and w_i can be conveniently obtained based on the tabulated nodes and weights of Gaussian quadrature formula [21], which are shown in **Table 2**, where l_i^{G-H} and ω_i^{G-H}, l_i^{G-La} and ω_i^{G-La}, l_i^{G-Le} and ω_i^{G-Le}, respectively, represent the quadrature nodes and weights of Gaussian-Hermite, Gaussian-Laguerre, and Gaussian-Legendre quadrature formula; λ is the parameter of exponential distribution; and μ_1 and μ_0 denote the lower and upper bounds of uniform distribution.

Normal		Exponential		Uniform	
l_i	ω_i	l_i	ω_i	l_i	ω_i
$\sqrt{2}\,\sigma l_i^{G-H}+\mu$	$\frac{\omega_i^{G-H}}{\sqrt{\pi}}$	$\frac{l_i^{G-La}}{\lambda}$	ω_i^{G-La}	$\frac{\mu_1-\mu_0}{2} l_i^{G-Le}+\frac{\mu_1+\mu_0}{2}$	$\frac{\omega_i^{G-Le}}{2}$

Table 2. l_i and ω_i calculated based on Gaussian quadrature.

However, the distributions of random inputs may not follow the Askey scheme, or are even nontrivial, or even exist in some raw data with a cumulative histogram of complicated shapes. Thus, such way to derive these nodes and weighs is not applicable in this case. In this work, a simple method is proposed based on the moment-matching equations below to obtain the one-dimensional quadrature nodes and weights.

$$\begin{aligned}
&\omega_0+\omega_1+\cdots+\omega_n=\int_{x\in\Omega}1d\Gamma(x)\\
&\omega_0 l_0+\omega_1 l_1+\cdots+\omega_n l_n=\int_{x\in\Omega}xd\Gamma(x)\\
&\vdots\\
&\omega_0(l_0)^n+\omega_1(l_1)^n+\cdots+\omega_n(l_n)^n=\int_{x\in\Omega}x^r d\Gamma(x)
\end{aligned}\tag{13}$$

where l_i and ω_i (i = 0, 1, …, n) are respectively the ith one-dimensional Gaussian quadrature nodes and weights, which theoretically can be obtained by solving Eq. (13).

However, Eq. (13) are multivariate nonlinear equations, which are difficult to solve when the number of equations is large (n + 1 > 7). It is noted that the one-dimensional polynomial basis $P^{(k)}$ corresponding to each dimension constructed above is orthogonal. Therefore, its zeros are just the Gaussian quadrature nodes l_i, which can be easily obtained by solving $P^{(k)}$ = 0. Through substituting l_i into Eq. (13), the n + 1 weights ω_i can be conveniently calculated. To calculate Eq. (13) of PCE order p, generally at least p + 1 one-dimensional nodes should be generated to ensure the accuracy, i.e., $n \geq p$, which means that 0 to at least pth statistic moments of the random variable $\boldsymbol{X}$ should be matched. In this work, n is set as $n = p$.

In the same way, the nodes and weights in other dimensions are obtained conveniently. Then, the numerator can be calculated by the full factorial numerical integration (FFNI) method [8] for lower-dimensional problems (d < 4) as

$$E[y\,\Phi_i(\boldsymbol{X})]=\mathrm{E}[Z(\boldsymbol{X})]\approx\sum_{i_1=1}^{m_1}\omega_{i_1}\cdots\sum_{i_j=1}^{m_j}\omega_{i_j}\cdots\sum_{i_d=1}^{m_d}\omega_{i_d}Z(l_{i_1},\cdots,l_{i_j},\cdots,l_{i_d})=\sum_{j=1}^{N}W_j Z(L_j)\tag{14}$$

where l_{i_j} and ω_{i_j}, respectively, represent the one-dimensional nodes and weights of the jth random input variable, which can be obtained using the way introduced above; L_i and W_i (i = 1, …, N) are the d-dimensional nodes and weights, respectively.

Generally, m is set as $m \geq p + 1$ (p is the order of the PCE model). If the number of nodes N for calculating $\mathrm{E}[y\Phi_i(\boldsymbol{X})]$ is too small, which is not matched with the PCE order, large error would

http://dx.doi.org/10.5772/intechopen.68484

be induced. Therefore, the conclusion that the higher the PCE order, the more accurate the UP results is based on the fact that $E[y\Phi_i(\boldsymbol{X})]$ has been calculated accurately enough. Clearly, the number of nodes N is increased exponentially with the increase of dimension d, causing curse of dimensionality. Therefore, FFNI is only suitable for lower-dimensional problems ($d < 4$).

For higher-dimensional problems ($d \geq 4$), the sparse grid numerical integration method [22] can be used to calculate $E[y\Phi_i(\boldsymbol{X})]$ to reduce the computational cost:

$$E[y\Phi_i(\boldsymbol{X})] = E[Z(\boldsymbol{X})] \approx \sum_{q-d+1 \leq |i| \leq q} (-1)^{q-|i|} \binom{d-1}{q-|i|} (\omega_{i_1} \ldots \omega_{i_j} \ldots \omega_{i_d})\, Z(l_{i_1}, \cdots, l_{i_j}, \cdots, l_{i_d}) \tag{15}$$

where $|i| = i_1 + , \ldots, + i_d$ and $i_1, \ldots, i_d$ are the accuracy index corresponding to each dimension.

For the FFNI-based method, if m nodes are selected on each dimension ($m_1 = \ldots = m_d = m$), $2m - 1$ accuracy level can be obtained. For the sparse grid-based method, $2k + 1$ accuracy level can be obtained with the accuracy level $k = q - d$. For example, if $k = 2$ and $d = 8$, for the sparse grid-based method, 17 nodes are required yielding 5th ($2\times2 + 1$)-order accuracy level. For the FFNI-based method, to obtain the same accuracy level 5 ($2\times3 - 1$), m should be $m = 3$ requiring 3^8 nodes. Clearly, to obtain the same accuracy level, the number of nodes of the sparse grid-based method is much smaller than that of the FFNI-based method if d is relatively large.

In this chapter, we focus on extending the Galerkin projection to the DD-PCE method to address higher-dimensional UP problems and then exploring the relative merits of these PCE approaches. For the case with only small data sets, both DD-PCE and the existing distribution-based method (gPCE) may produce large errors for UP, and the estimation of PDF for the existing PCE methods is problem dependent and very subjective. It is difficult to make a comparison effectively between DD-PCE and the existing PCE methods. Therefore, during the comparison, it is assumed that there are enough data of the random input to ensure the accuracy of the moments.

2.3. Comparative study of various PCE methods

In this section, the enhanced DD-PCE method, the recognized gPCE method, and the GS-PCE method that can address arbitrary random distributions are applied to uncertainty propagation to calculate the first four statistic moments (mean μ, standard deviation σ, skewness β_1, kurtosis β_2) and probability of failure (P_f), of which the results are compared to help designers to choose the most suitable PCE method for UP. To comprehensively compare the three PCE approaches, four cases are respectively tested on four mathematical functions with varying nonlinearity and dimension shown in **Table 3** and *N, U, Exp, Wbl, Rayl*, and *Logn* denote normal, uniform, exponential, Weibull, Rayleigh, and lognormal distribution, respectively. P_f is defined as P_f = probability ($y \leq 0$).

The PCE order is set as $p = 5$ for all the functions for comparison, which means that 0–9th statistic moments of the random inputs should be matched to construct the one-dimensional orthogonal polynomials for the DD-PCE approach. For the first and second functions, FFNI-based Galerkin projection is used to calculate the PCE coefficients, while for the latter two,

Function 1: $y = x_1+x_2+x_3$
Case 1: $x_1 \sim U(1,2)$, $x_2 \sim N(1,0.2)$, $x_3 \sim Exp(0.5)$
Case 2: $x_1 \sim Wbl(2,6)$, $x_2 \sim Rayl(3)$, $x_3 \sim Logn(0,0.25)$
Case 3: x_1~BD, x_2~ BD, $x_3 \sim N(0,0.2)$
Case 4: 500 and 10^7 sample points x_1~BM, x_2~ BM, $x_3 \sim N(-0.8,0.2)$
Function 2: $y = \sin(x_1) - \cos^2(x_2) + x_3\sin(x_1) + 0.9$
Case 1: $x_1 \sim N(0.5,0.2)$, $x_2 \sim U(0,1.5)$, $x_3 \sim Exp(0.1)$
Case 2: $x_1 \sim Wbl(2,3)$, $x_2 \sim Rayl(0.2)$, $x_3 \sim Logn(0,0.25)$
Case 3: x_1~ BD, x_2~ BD, $x_3 \sim U(0,1)$
Case 4: 500 and 10^7 sample points x_1~BM, x_2~ BM, $x_3 \sim U(0.4,2)$
Function 3: $y = e^{-x1}\cos(x_2) + x_3e^{-x4x5} - e^{-x6}$
Case 1: $x_1 \sim N(1,0.2)$, $x_2 \sim U(-1,1)$, $x_3 \sim N(1,0.2)$, $x_4 \sim U(-1,1)$, $x_5 \sim N(0,0.2)$, $x_6 \sim U(0,2)$
Case 2: $x_1 \sim Wbl(1,5)$, $x_2 \sim Rayl(0.5)$, $x_3 \sim Logn(0.5,0.25)$,$x_4 \sim Rayl$ (0.3), $x_5 \sim Wbl(1,5)$, $x_6 \sim Rayl(1)$
Case 3: x_1~ *BD*, x_2~ *BD*, x_3~ $N(2,0.2)$, $x_4 \sim U(-1,0)$, $x_5 \sim N(1,0.2)$, $x_6 \sim U(-1,4)$
Case 4: 500&10^7sample points x_1~*BM*, x_2~*Rayl* (0.3), x_3~*BM* ,x_4 ~*Rayl* (0.3), x_5 ~*BM*, x_6 ~*Rayl* (1)
Function 4: $y = x_1^2x_2^2 - x_3^2x_4^2 + x_5^2x_6^2 - x_7^2x_8^2 + x_9^2x_{10}^2$
Case 1: $x_1 \sim N(1,0.2)$, $x_2 \sim U(0,2)$, $x_3 \sim N(0,0.2)$, $x_4 \sim U(0,2)$, $x_5 \sim N(1,0.2)$, $x_6 \sim U(0,2)$, $x_7 \sim N(0,0.2)$, $x_8 \sim U(0,2)$, $x_9 \sim N(1,0.2)$, $x_{10} \sim U(0,2)$
Case 2: $x_1 \sim Wbl(1,5)$, $x_2 \sim Rayl(1)$, $x_3 \sim Wbl(1,5)$, $x_4 \sim Rayl(0.3)$, $x_5 \sim Wbl(1,5)$, $x_6 \sim Rayl(1)$, $x_7 \sim Wbl(1,5)$, $x_8 \sim Rayl(0.3)$, $x_9 \sim Wbl(1,5)$, $x_{10} \sim Rayl(1)$
Case 3: $x_1 \sim N(1,0.2)$, $x_2 \sim N(1,0.2)$, $x_3 \sim BD$, $x_4 \sim BD$, $x_5 \sim N(1,0.2)$, $x_6 \sim N(1,0.2)$, $x_7 \sim BD$, $x_8 \sim BD$, $x_9 \sim N(1,0.2)$, $x_{10} \sim N(1,0.2)$
Case 4: 500 and 10^7 sample points $x_1 \sim N(1.5,0.2)$, $x_2 \sim N(1,0.2)$, $x_3 \sim BM$, $x_4 \sim BM$, $x_5 \sim N(1,0.2)$, $x_6 \sim N(1,0.2)$, $x_7 \sim N(0,0.2)$, $x_8 \sim N(0,0.2)$, $x_9 \sim N(1,0.2)$, $x_{10} \sim N(1,0.2)$

Table 3. Test functions and random input information of four cases.

the sparse grid-based method with accuracy level $k = 4$ is used since the dimension is higher. The results of MCS with 10^7 runs are used to benchmark the effectiveness of the three methods.

In Case 1, all the random input distributions are known and belong to the Askey scheme. The test results are shown in **Tables 4–7**, where the bold numbers with underline are the relatively best results and e represents the relative errors of the first four moments (μ, σ, β_1, β_2) with respect to MCS. P_f estimated by MCS is presented with 95% confidence interval. The results marked with * are from the sparse grid-based method. From these tables, it is found that with the same number of function calls (denoted as *Ns*), DD-PCE, gPCE, and GS-PCE produce almost the same results of the statistic moments, which are very similar to those of MCS (with the largest error as 2.6927%). The estimation of P_f for all the methods is within the 95% confidence interval with respect to MCS, indicating the high accuracy of UP. Although the orthogonal polynomial basis for DD-PCE is constructed by matching only 0–9th statistic moments of the random input variable instead of the complete PDFs for gPCE and GS-PCE, the results are accurate enough in this case. Moreover, the application of sparse grid technique to DD-PCE can greatly reduce the function calls for higher-dimensional problems (see **Tables 5** and **6**), while

Methods	MCS	DD-PCE	*g*PCE	GS-PCE
e_μ (%)	–	0.0050	**0**	0.0100
e_σ (%)	–	0.0164	(0.0164)	0.0164
e_{β_1} (%)	–	0.1367	0.1367	**0.1094**
e_{β_2} (%)	–	0.4877	0.3032	**0.2199**
P_f (1e^{-3})	[8.5185,8.6328]	8.5472	8.5901	8.5688
N_s	10^7	125	125	125

Table 4. Results of function 1 (Case 1).

Methods	MCS	DD-PCE	*g*PCE	GS-PCE
e_μ (%)	–	0.0115	**0**	0.0231
e_σ (%)	–	0.0516	**0**	0.0258
e_{β_1} (%)	–	**0**	0.4202	5.4852
e_{β_2} (%)	–	0.1725	**0.0814**	0.0958
P_f (1e^{-3})	[3.1403,3.2101]	3.1713	3.2017	3.1936
N_s	10^7	125	125	125

Table 5. Results of function 2 (Case 1).

Methods	MCS	DD-PCE*	*g*PCE*	GS-PCE*
e_μ (%)	–	**0**	0.0112	0.0225
e_σ (%)	–	0.0288	0.0288	(0.0288
e_{β_1} (%)	–	2.2284	2.6927	**1.7642**
e_{β_2} (%)	–	0.6040	0.6074	**0.5028**
P_f (1e^{-3})	[4.8454,4.9318]	4.8993	4.8669	4.9074
N_s	10^7	1820	1820	1820

Table 6. Results of function 3 (Case 1).

Methods	MCS	DD-PCE*	*g*PCE*	GS-PCE*
e_μ (%)	–	0.0123	0.0296	**0.0074**
e_σ (%)	–	**0.0402**	0.0723	0.0522
e_{β_1} (%)	–	0.1018	**0.0890**	0.1399
e_{β_2} (%)	–	0.1050	**0**	0.1326
P_f (1e^{-3})	[4.2476,4.3286]	4.2627	4.2881	4.2562
N_s	10^7	10,626	10,626	10,626

Table 7. Results of function 4 (Case 1).

exhibiting good accuracy. Especially for the fourth function, with FFNI, the computational cost is very large (N_s = 976,562).

In Case 2, all the random input distributions are known but do not belong to the Askey scheme. In this case, the Rosenblatt transformation is employed for the gPCE method first. However, DD-PCE and GS-PCE can be directly used. The results are shown in **Tables 8–11**. It is observed that overall DD-PCE and GS-PCE perform better than gPCE, yielding results that are close to those of MCS. The reason is that the transformation in gPCE would induce error. Specifically, in **Tables 9** and **10**, the gPCE method causes relatively large errors due to the transformation. In addition, note the numbers with shadow, they are clearly larger than those

Methods	**MCS**	**DD-PCE**	***g*PCE**	**GS-PCE**
e_μ (%)	–	0.0196	**0.0087**	0.0175
e_σ (%)	–	0.0298	**0.0099**	0.0199
e_{β_1} (%)	–	0.2573	0.2059	**0.2059**
e_{β_2} (%)	–	0.2170	0.2263	**0.0899**
P_f (1e^{-4})	[1.9818,2.1602]	2.0360	2.1490	2.0480
N_s	10^7	125	125	125

Table 8. Results of function 1 (Case 2).

Methods	**MCS**	**DD-PCE**	***g*PCE**	**GS-PCE**
e_μ (%)	–	0.0243	0.0182	**0.0061**
e_σ (%)	–	0.0467	0.2101	**0**
e_{β_1} (%)	–	**1.8877**	8.0227	2.5956
e_{β_2} (%)	–	0.0307	1.1659	**0.0279**
P_f (1e^{-4})	[9.0052,9.3808]	9.0130	7.9720	9.0250
N_s	10^7	125	125	125

Table 9. Results of function 2 (Case2).

Methods	**MCS**	**DD-PCE***	***g*PCE***	**GS-PCE***
e_μ (%)	–	**0**	0.0084	**0**
e_σ (%)	–	**0.0443**	0.0887	**0.0443**
e_{β_1} (%)	–	**0.3471**	0.6480	0.4397
e_{β_2} (%)	–	**0.0419**	0.1927	0.1368
P_f (1e^{-3})	[1.0859,1.1271]	1.0963	1.2291	1.1188
N_s	10^7	1820	1820	1820

Table 10. Results of function 3 (Case2).

Methods	MCS	DD-PCE*	*g*PCE*	GS-PCE*
e_μ (%)	–	0.0240	**0.0180**	0.0320
e_σ (%)	–	**0.0111**	0.0722	0.0250
e_{β_1} (%)	–	0.2170	**0.1979**	0.2362
e_{β_2} (%)	–	**0.4229**	1.9117	0.4582
P_f (1e^{-3})	[4.4019,4.4843]	4.4635	4.6942	4.4200
N_s	10^7	10,626	10,626	10,626

Table 11. Results of function 4 (Case 2).

of DD-PCE and GS-PCE, and P_f is outside the range of the 95% confidence interval of MCS. The interpretation is that since the first function is linear, the impact of transformation employed in gPCE on the accuracy of UP is small; while, for the second and third functions, they are more complicated and nonlinear (including trigonometric and exponential terms), the error induced by the transformation employed in gPCE is amplified more. The fourth function is a nonlinear polynomial one, which is easier to be handled than functions 2 and 3 in doing UP. Therefore, the results are generally accurate except P_f that is still outside the range of the 95% confidence interval of MCS. Moreover, the application of sparse grid greatly reduces N_s, exhibiting good potential applications for higher-dimensional problems.

In Case 3, the PDFs of some variables is bounded (BD) as below,

$$f(x) = \begin{cases} 2x, 0 < x < 1 \\ 0, \text{otherwise} \end{cases} \tag{16}$$

and the rest of the variables follow typical distributions. In this case, the Rosenblatt transformation is also employed for the gPCE method first.

From the results in **Tables 12–15**, it is found that generally large errors are induced by gPCE, especially the numbers with shadow in the tables. Since the first two variables follow the distribution bounded in an interval, the error induced by the transformation is large and all values of P_f are outside the confidence intervals for gPCE. While, the results of DD-PCE and GS-PCE are generally accurate and comparable, which are still very close to those of MCS. It should be noted that although the error of gPCE is the largest, all P_f by the three methods are

Methods	MCS	DD-PCE	*g*PCE	GS-PCE
e_μ (%)	–	**0**	0.0150	**0**
e_σ (%)	–	0.0195	24.1063	**0**
e_{β_1} (%)	–	0.1359	36.9565	**0.1132**
e_{β_2} (%)	–	**0.0545**	12.3239	**0.0545**
P_f (1e^{-3})	[4.9841,5.0717]	5.0038	5.2620	5.0333
N_s	10^7	125	125	125

Table 12. Results of function 1 (Case 3).

Methods	MCS	DD-PCE	*g*PCE	GS-PCE
e_μ (%)	–	**0.0083**	0.0914	**0.0083**
e_σ (%)	–	**0.0213**	19.7662	**0.0213**
e_{β_1} (%)	–	0.4186	123.2093	**0.3256**
e_{β_2} (%)	–	0.0555	12.7841	**0.0476**
P_f (1^{e-3})	[1.4429,1.4903]	1.4449	1.7890	1.4452
N_s	10^7	125	125	125

Table 13. Results of function 2 (Case 3).

Methods	MCS	DD-PCE*	*g*PCE*	GS-PCE*
e_μ (%)	–	**0.0359**	0.7473	0.0598
e_σ (%)	–	**0.3983**	(4.2798	0.3693
e_{β_1} (%)	–	0.1221	22.5570	0.2036
e_{β_2} (%)	–	0.6186	77.1134	0.6321
P_f ($1e^{-3}$)	[3.1972,3.2676]	***2.6222***	***8.9269***	***2.6071***
N_s	10^7	1820	1820	1820

Table 14. Results of function 3 (Case 3).

Methods	MCS	DD-PCE*	*g*PCE*	GS-PCE*
e_μ (%)	–	**0.0039**	6.4980	0.0194
e_σ (%)	–	0.0409	8.2618	**0.0164**
e_{β_1} (%)	–	0.1187	50.3681	**0.0475**
e_{β_2} (%)	–	**0.1720**	11.8984	0.1949
P_f ($1e^{-3}$)	[8.6089,8.7237]	8.6559	0.8227	8.6728
N_s	10^7	10,626	10,626	10,626

Table 15. Results of function 4 (Case 3).

outside the confidence intervals for function 3 (italic numbers) since this function is the most nonlinear and complicated. Hence, we increase the PCE order p and accuracy level k of the sparse grid to $p = 6$ and $k = 5$, and the results of P_f for DD-PCE, gPCE, and GS-PCE are 3.1263, 3.1446, and 3.1350, exhibiting evident improvement. Clearly with the same *Ns*, DD-PCE and GS-PCE are much more accurate than gPCE when nontrivial distribution is involved. These results further demonstrates the effectiveness and advantage of the enhanced DD-PCE for UP.

In Case 4, the distributions of the random input variables are unknown and only some data exist. Although, based on the data, the analytical PDF can be obtained through some experience systems, such as Johnson or Pearson system [8], if the distribution of the data is very

http://dx.doi.org/10.5772/intechopen.68484

complicated, such as with a complicated cumulative histogram of bi- or multimodes, it is often very difficult to obtain the analytical PDF accurately. As is well-known that the Pearson system based on the first four statistic moments of the random variable would produce large errors for bimode (BM) or multimode PDFs. Evidently, the existing PCE approaches, including gPCE and GS-PCE, may produce large errors since they all depend on the exact PDFs of the random inputs in this case. However, DD-PCE can still work since it is a data-driven approach. To explore the effectiveness and advantage of DD-PCE over the other two approaches, it is assumed that the input data for some random input variables have a complicated bimode (BM) histogram shown in **Figure 3** and the data for the rest from the typical distributions. Therefore, for the convenience and effectiveness of test, all the input data are generated based on the PDFs, of which the PDF of BM distribution is shown in Eq. (17). It should be pointed out that the PDFs actually are unknown and only some data exist in practice.

$$f_{\mathrm{PDF}} = \frac{0.647}{0.1\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\times 0.1^2}\right) + \frac{0.353}{0.2\sqrt{2\pi}} \exp\left(-\frac{(x-1)^2}{2\times 0.2^2}\right), x \in [-\infty, +\infty] \tag{17}$$

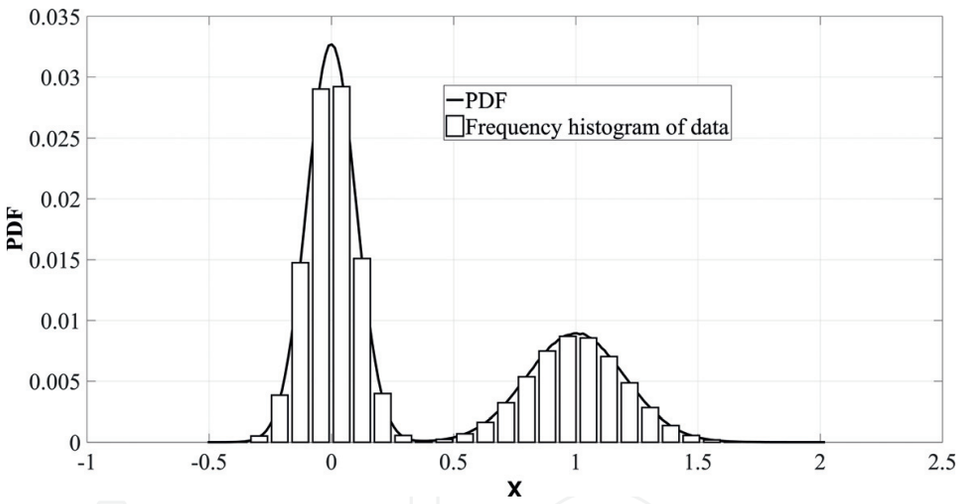

Figure 3. PDF plot of the bimodal distribution.

We tested small (500) and large (10^7) numbers of input data to investigate the impact of number of data on the accuracy of UP. The results are shown in **Tables 16–19**, from which it is noticed that the results of DD-PCE are generally very close to those of MCS when the number of sample points of the random input variables is large (10^7). When only 500 sample points are used, the errors are much larger. It means that the accuracy of DD-PCE is improved with the increase of the number of sample points. The reason is very simple that with the increase of the number of sample points, the statistic moments of random input variables calculated are more accurate, which would undoubtedly increase the accuracy of UP. The observation exhibits great agreements to what has been reported in work of Oladyshkin and Nowak. Similar to Case 3, the estimated P_f is outside the confidence intervals for function 3 since this function is the most nonlinear and the random distribution is more irregular, which can be improved by

Methods	MCS	DD-PCE (10^7)	DD-PCE (500)
e_μ (%)	–	0.0066	1.4873
e_σ (%)	–	0.0196	0.0688
e_{β_1} (%)	–	0.0150	0.0451
e_{β_2} (%)	–	0.0052	3.2327
P_f ($1e^{-3}$)	[1.4772,1.5252]	1.5069	0
N_s	10^7	125	125

Table 16. Results of function 1 (Case 4).

Methods	MCS	DD-PCE(10^7)	DD-PCE(500)
e_μ (%)	–	0.0132	0.4350
e_σ (%)	–	0.0109	0.1957
e_{β_1} (%)	–	0.1159	13.4783
e_{β_2} (%)	–	0.0131	0.8956
P_f ($1e^{-3}$)	[6.4478,6.5474]	6.4703	8.000
N_s	10^7	125	125

Table 17. Results of function 2 (Case 4).

Methods	MCS	DD-PCE(10^7)	DD-PCE(500)
e_μ (%)	–	0.0327	0.6047
e_σ (%)	–	2.7503	5.3717
e_{β_1} (%)	–	3.8373	9.5932
e_{β_2} (%)	–	0.5563	1.3573
P_f ($1e^{-3}$)	[7.7830,7.8924]	6.6667	6.0000
N_s	10^7	1820	1820

Table 18. Results of function 3 (Case 4).

Methods	MCS	DD-PCE(10^7)	DD-PCE(500)
e_μ (%)	–	0.0024	0.1925
e_σ (%)	–	0.0241	3.5156
e_{β_1} (%)	–	0.4149	214.4537
e_{β_2} (%)	–	0.0170	11.9346
P_f ($1e^{-3}$)	[9.2650,9.3842]	9.2937	0
N_s	10^7	10626	10626

Table 19. Results of function 4 (Case 4).

http://dx.doi.org/10.5772/intechopen.68484

increasing N_s. This means that the generally the more nonlinear the function and the more irregular the random input distribution, the more difficult it is to achieve accurate UP results. These results further demonstrate the effectiveness and advantage of the enhanced DD-PCE method for UP.

To study the convergence property of the enhanced DD-PCE method, the errors (e) of moments and P_f with different PCE orders obtained by the proposed one as well as gPCE and GS-PCE are shown in **Figures 4–7**, taking Function 2, for example. Clearly, similar to the existing two methods, with the increase of the PCE order, the errors decrease significantly, exhibiting an approximate exponential convergence rate. Meanwhile, it is observed that the speed of convergence in Case 1 (Askey scheme) is the fastest. Generally, the more irregular the input distribution and the more nonlinear the function, the slower is the convergence process. In addition, it is also

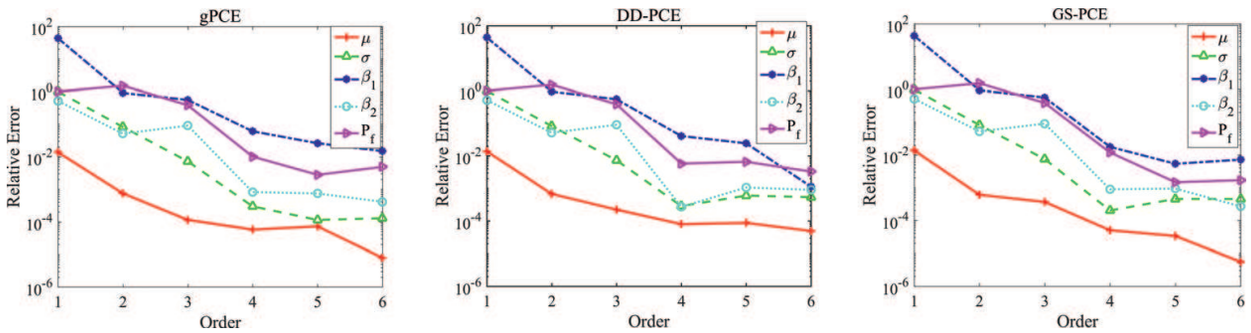

Figure 4. Errors with respect to different PCE orders (Case 1).

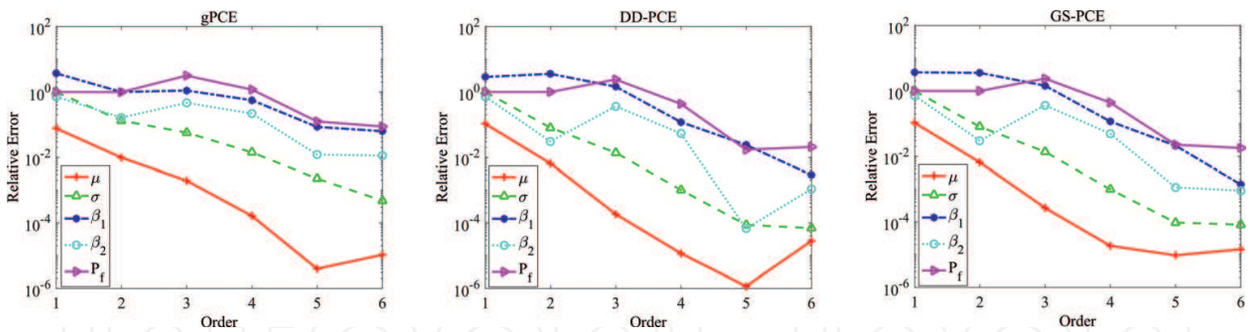

Figure 5. Errors with respect to different PCE orders (Case 2).

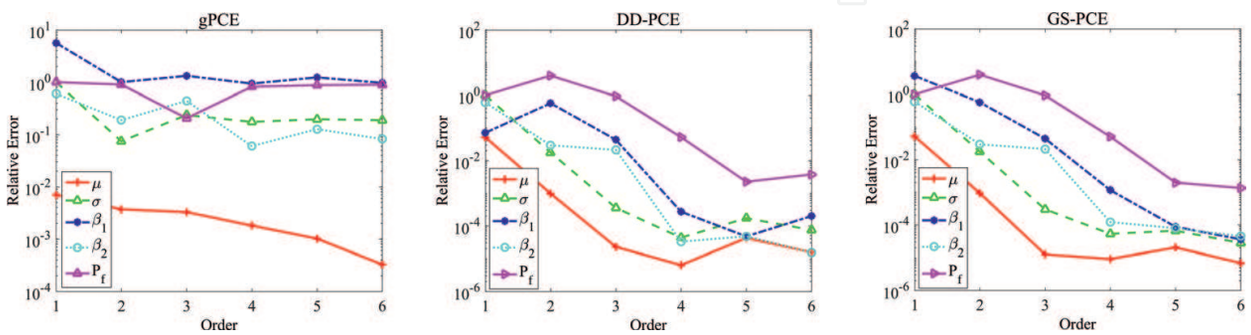

Figure 6. Errors with respect to different PCE orders (Case 3).

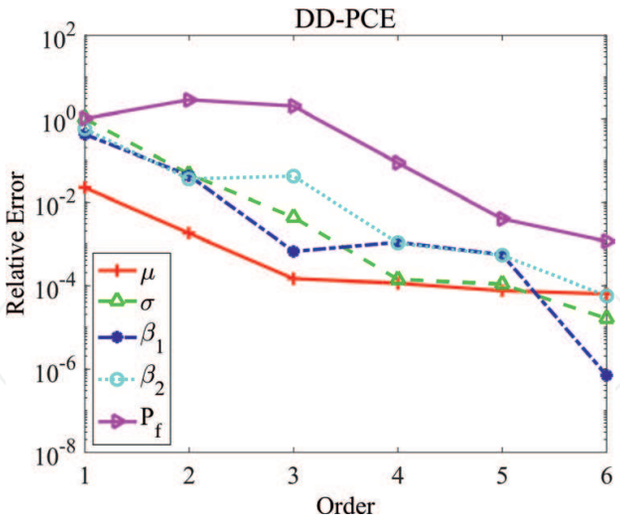

Figure 7. Errors with respect to different PCE orders (Case 4).

noticed that for Case 3, since x_1 and x_2 follow the nontrivial distribution, the convergence rate is very slow for gPCE (see left in **Figure 6**) due to the error induced by the transformation.

2.4. Summary

Overall, the three approaches produce comparably good results when the random inputs follow the Askey scheme. However, gPCE is the most mature and convenient to be implemented since there is no need to construct the orthogonal polynomials. When the PDFs of random inputs are unknown but do not follow the Askey scheme, large errors would be induced by the transformation for gPCE and the rest two PCE methods are comparable in accuracy and implementation complexity. It should also be pointed out that for DD-PCE, when constructing one-dimensional polynomials, the statistic moments (often 0–10 order) should be calculated first. If large gap exists between the high-order and low-order moments, the matrix singularity would happen in solving the linear equations (Eq. (7)). Therefore, in this case, GS-PCE is preferable especially when the function is highly nonlinear. When the PDF is unknown and cannot be obtained accurately, such as when random inputs exist as some raw data with a complicated cumulative histogram, only the DD-PCE method can still perform well since it is a data-driven method instead of the probabilistic-distribution-driven, while large errors would be produced if GS-PCE and gPCE are employed. However, more efforts should be made to solve the numerical problems in the DD-PCE method to make it more robust and applicable in constructing the one-dimensional orthogonal polynomials.

3. A sparse data-driven PCE method

The size of the truncated polynomial terms in the full PCE model is increased with the increase of the dimension of random inputs d and the order of PCE model p (see Eq. (1)), resulting in a significant growth of the computational cost. Therefore, attempts are made in this section on the full DD-PCE method introduced in Section 2 to reduce the computational cost. Accordingly, a sparse PCE approximation, which only contains a small number of polynomial terms compared

http://dx.doi.org/10.5772/intechopen.68484

to a classical full representation, is eventually provided by using the least angle regression (LAR) theory [23] and the sequential sampling method. The original LAR method is used for variables selection, aiming to find the most important variables with respect to a function response. In this work, LAR is extended to select some polynomial terms $\Phi_i(\boldsymbol{x})$ from the full PCE model that have the greatest impact on the model response $y \approx M(\boldsymbol{x}) = \sum_{i=0}^{P} b_i \Phi_i(\boldsymbol{x})$ in a similar way.

Although the computational cost and accuracy are dependent on the PCE order, how to determine a suitable order that compromises between accuracy and efficiency is not within the scope of this chapter. In common situations, PCE of order $p = 2$ or 3 can produce results with good agreement to MCS for the output PDF estimation [24]. For more rigorous approaches of adaptively determining the order of the PCE model rather than specifying it in advance, readers can refer to references [25, 26].

3.1. Procedure of data-driven PCE method

A step-by-step description of the proposed method is given in detail as below with a side-by-side flowchart in **Figure 8**.

Step 1. Given the information of the random inputs (raw data or probabilistic distributions), specify the PCE order p, and then construct the full DD-PCE model without computing the PCE coefficients.

Step 2. Generate the initial input sample points $\boldsymbol{X} = [\boldsymbol{x}_1,\ldots,\boldsymbol{x}_m,\ldots,\boldsymbol{x}_N]^T$ according to the distributions of the random inputs or select the sample points from the given raw data, where $\boldsymbol{x}_m = [\boldsymbol{x}_{m1}, \ldots,\boldsymbol{x}_{md}]$. Meanwhile, calculate the corresponding real function responses $\boldsymbol{y} = [y_1,\ldots,y_m,\ldots,y_N]^T$.

$\boldsymbol{X}$ is standardized to have mean 0 and unit length, and that the response $\boldsymbol{y}$ has mean 0.

$$\frac{1}{N}\sum_{m=1}^{N} x_{\mathrm{mn}} = 0, \quad \sqrt{\sum_{m=1}^{N} x_{\mathrm{mn}}^2} = 1 \quad (n = 1, \ldots, d), \quad \frac{1}{N}\sum_{m=1}^{N} y_m = 0 \tag{18}$$

Then one has all the standardized data as

$$\overline{\boldsymbol{X}} = \begin{bmatrix} \overline{x}_{11}, \overline{x}_{12}, \ldots, \overline{x}_{1d} \\ \overline{x}_{21}, \overline{x}_{22}, \ldots, \overline{x}_{2d} \\ \cdots \quad \cdots \\ \overline{x}_{N1}, \overline{x}_{N2}, \ldots, \overline{x}_{Nd} \end{bmatrix}, \overline{\boldsymbol{y}} = (\overline{y}_1, \ldots, \overline{y}_N)^T \tag{19}$$

Step 3. Set the iteration number as $K = 0$ and compute the values of all polynomial terms $\Phi_i(\boldsymbol{x})$ ($i = 0, 1, \ldots, P$) of the full PCE model in Eq. (1) by, respectively, substituting each input sample point $\boldsymbol{x}_m$ into them. Then one obtains the information matrix as

$$\Phi = \begin{bmatrix} \Phi_0(\boldsymbol{x}_1) & \Phi_1(\boldsymbol{x}_1), \ldots, & \Phi_P(\boldsymbol{x}_1) \\ \vdots\ldots & & \vdots \\ \Phi_0(\boldsymbol{x}_N) & \Phi_1(\boldsymbol{x}_N), \ldots, & \Phi_P(\boldsymbol{x}_N) \end{bmatrix} \tag{20}$$

Step 4. The LAR algorithm is employed to automatically detect some number (often $K + 1$) of significant orthogonal polynomial terms from the first $K + 1$ terms $\Phi_i(\boldsymbol{x})$ ($i = 0, 1, \ldots, K$) in Eq. (1), which will be retained to construct a sparse candidate PCE model that has a smaller scale than the full PCE model. For the introduction of the original LAR algorithm, readers can refer to reference [23] for more details.

Step 5. To save the computational cost, the leave-one-out cross-validation method [27] is employed to evaluate the accuracy of the candidate sparse PCE model constructed above, which is represented as the leave-one-out error analytically as below:

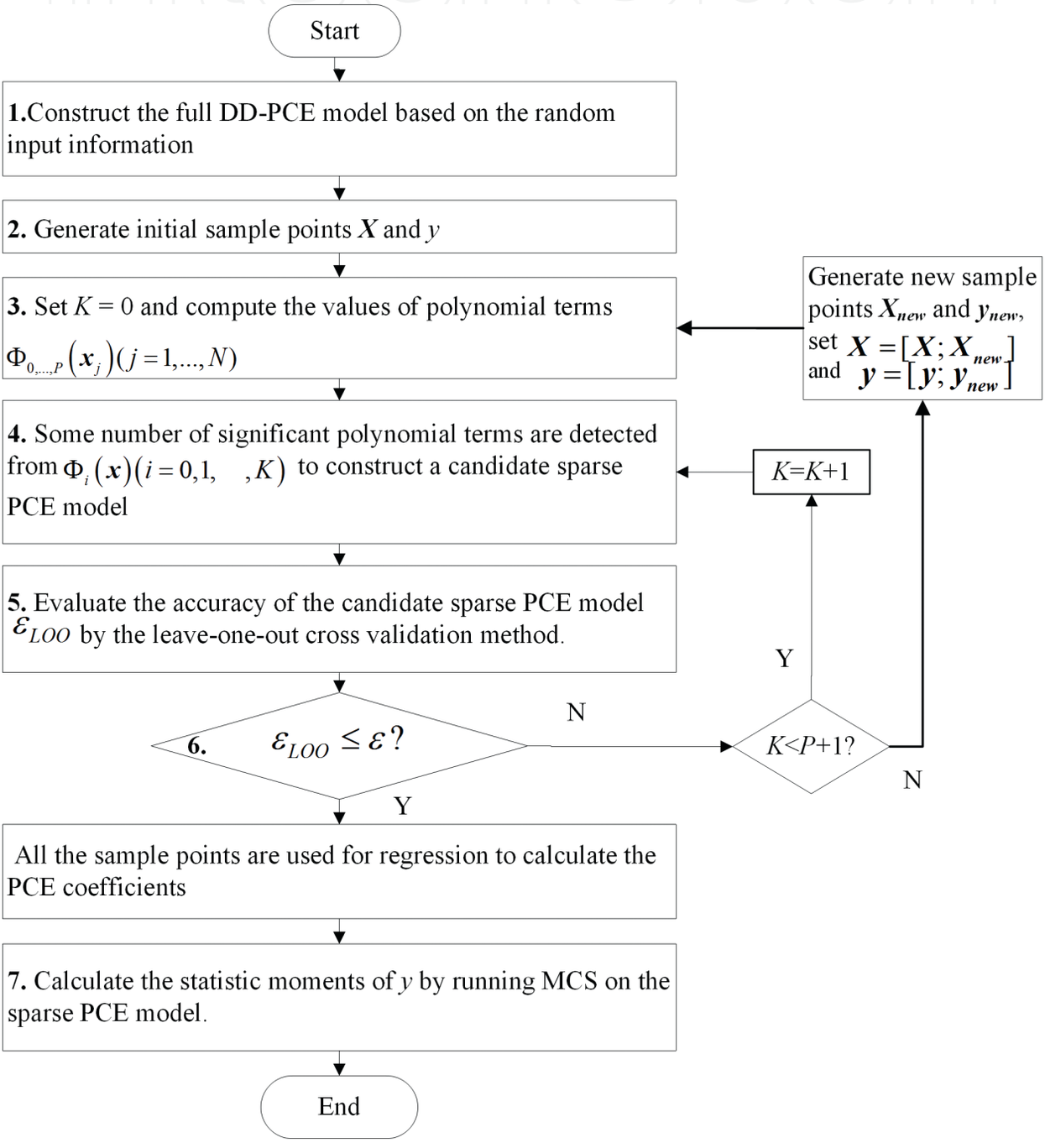

Figure 8. The flowchart of the sparse DD-PCE method.

http://dx.doi.org/10.5772/intechopen.68484

$$Err_{\mathrm{LOO}} = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{g(\mathbf{x}_j) - \hat{g}_{\boldsymbol{I}}^{(-j)}(\mathbf{x}_j)}{1-h_j}\right)^2 \tag{21}$$

where $g(\boldsymbol{x}_j)$ is the response value from the original model at the sample point $\boldsymbol{x}_j$; $\hat{g}_{\boldsymbol{I}}^{(-j)}$ represents the candidate sparse PCE model comprised of all the selected polynomial terms, of which the indices are stored in $\boldsymbol{I}$; the PCE coefficients of $\hat{g}_{\boldsymbol{I}}^{(-j)}$ are computed through using the ordinary least-square regression method based on the leave-one-out approach, i.e., the sample points for regression are $\boldsymbol{X}^{(-j)} = [\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{j-1}, \boldsymbol{x}_{j+1},\ldots,\boldsymbol{x}_N]^T$ and $\boldsymbol{y}^{(-j)} = [y_1,\ldots,y_{j-1}, y_{j+1},\ldots,y_N]^T$.

Once the PCE coefficients are calculated, the predicted value by the candidate sparse PCE model at the sample point $\boldsymbol{x}_j$ is calculated as $\hat{g}_{\boldsymbol{I}}^{(-j)}(\mathbf{x}_j)$; h_j is the jth diagonal element of the matrix $\boldsymbol{\Phi}_A(\boldsymbol{\Phi}^T{}_A\boldsymbol{\Phi}_A)^{-1}\boldsymbol{\Phi}^T{}_A$, where $\boldsymbol{\Phi}_A$ is a $N \times k$ matrix comprised of all the selected column vectors $\boldsymbol{\Phi}_i = [\Phi_i(\boldsymbol{x_1}), \ldots, \Phi_i(\boldsymbol{x}_N)]^T$ ($i \in I$) and k is the number of selected polynomial terms.

To evaluate the accuracy more effectively, the relative error is employed based on Err_{LOO} as

$$\varepsilon_{\mathrm{LOO}} = Err_{\mathrm{LOO}}/\hat{V}(\mathbf{y}) \tag{22}$$

where $\hat{V}(\mathbf{y})$ denotes the empirical variance of the response sample points $\boldsymbol{y}$, which is calculated by

$$\hat{V}(\mathbf{y}) = \frac{1}{N-1}\sum_{j=1}^{N}(y_j - \overline{y})^2,\ \overline{y} = \frac{1}{N}\sum_{j=1}^{N}y_j \tag{23}$$

Step 6. Check the stop criterion:

If the accuracy $\varepsilon_{\mathrm{LOO}}$ satisfies the target threshold ε, i.e., $\varepsilon_{\mathrm{Loo}} \le \varepsilon$, the procedure will be stopped, the PCE model obtained by LAR in Step 4 will be considered as the final one, and all the sample points will be used for regression to calculate the PCE coefficients of the current sparse PCE model;

If $\varepsilon_{\mathrm{Loo}} > \varepsilon$ and $K < P$, set $K = K + 1$ and go to Step 4 to find another candidate sparse PCE model by LAR;

If $\varepsilon_{\mathrm{Loo}} > \varepsilon$ and $K = P$, generate some new sample points $\boldsymbol{X}_{\mathrm{new}}$ with the sequential sampling technique and calculate the corresponding responses $\boldsymbol{y}_{\mathrm{new}}$, and add the new sample points into the old ones as $\boldsymbol{X} = [\boldsymbol{X}; \boldsymbol{X}_{\mathrm{new}}]$ and $\boldsymbol{y} = [\boldsymbol{y}; \boldsymbol{y}_{\mathrm{new}}]$, then go to Step 3 to find another candidate sparse PCE model.

In this work, if the PDF of random input is known, a large number of sample points are generated as the database according to the PDF beforehand; if the PDF of random input is unknown, the raw data are considered as the database. Each sample point in the database has its own index. The initial sample points are selected from the database through randomly and uniformly generating their indices. Then these sample points will be removed from the database and the rest will be indexed again. Similarly, by randomly and uniformly generating the indices, the sequential sample points will be selected from the reduced database. By using this

sampling strategy, the sample points are distributed uniformly as far as possible, which is helpful to improve the accuracy of the PCE coefficient calculation.

Step 7. Based on the final sparse PCE model, the probabilistic properties of y can be obtained by running MCS or analytically.

3.2. Comparative study

In this section, the proposed sparse DD-PCE method (shortened as sDD-PCE hereafter) is applied to three mathematical examples to calculate the mean and variance of the output responses. The full DD-PCE (shortened as fDD-PCE hereafter) method that adopts a full PCE structure and one-stage sampling with the size of one times the number of PCE coefficients is also applied to UP, of which the results are compared to those of sDD-PCE to demonstrate its effectiveness and advantage.

The test examples of varying dimensions including their input information are shown in **Table 20**, in which the symbols $\mathcal{N}, \mathcal{U}$ and $\mathcal{E}$ respectively, denote normal, uniform, and exponential distribution. To fully explore the applicability of sDD-PCE, three different cases of the random input information that almost cover all the situations in practice (Case 1: raw data; Case 2: common distribution; Case 3: nontrivial distribution) are considered. The nontrivial bimodal distribution (denoted as $\mathcal{BD}$) used in Section 2.3 (Eq. (16)) is considered.

Another type of nontrivial distribution considered here is invented by conducting square operation on the sample points from some common distributions (see Case 3 in Function 2). The target accuracy ε of sDD-PCE is set as 10^{-5}. Meanwhile, to ensure the effectiveness of comparison between sDD-PCE and fDD-PCE, the order of the PCE model p is set as the same

---

Function 1: $f_1 = X_1X_2$

Case 1: 10^5 raw data

Case 2: X_1~$\mathcal{N}(1,0.2^2)$, X_2~$\mathcal{U}(0.4,1.6)$

Case 3: X_1 and X_2 ~$\mathcal{BD}$

Function 2: $f_2 = -X_1^2X_2^2 - 2X_3^4 + 3X_4^2 - 0.5X_5 + 4.5$

Case 1: 10^5 raw data

Case 2: X_1~$\mathcal{N}$ $(1,0.2^2)$, X_2~$\mathcal{U}$ $(0.4,1.6)$, X_3~$\mathcal{E}$ (0.1), X_4~$\mathcal{U}$ $(-0.5,1)$,X_5~$\mathcal{U}$ $(0.5,1)$.

Case 3: X_1~$\mathcal{BD}$, X_2~$\mathcal{U}$ (0.4,1.6).^2, X_3~$\mathcal{U}$ (0.5,1) .^2, X_4~$\mathcal{U}$ $(-0.5,1)$, X_5~$\mathcal{U}$ (0.5,1).

Function 3: $f_3 = -20\exp\left(-0.2\sqrt{\frac{1}{10}\sum_{i=1}^{10}x_i^2}\right) - \exp\left(\frac{1}{10}\sum_{i=1}^{10}\cos(2\pi x_i)\right)$

Case 1: 10^5 raw data

Case 2: X_1~$\mathcal{N}(1,0.2^2)$, X_2~$\mathcal{U}(0.4,1.6)$, X_3~$\mathcal{U}$ $(-1.5,15)$, X_4~$\mathcal{U}(-1,2)$, X_5~$\mathcal{U}(-15, 1)$, X_6~$\mathcal{N}(2,0.2^2)$, X_7~$\mathcal{U}(-3,3)$, X_8~$\mathcal{U}(-15,1.5)$, X_9~$\mathcal{U}(-2,15)$, X_{10}~$\mathcal{U}(-2,15)$.

Case 3: X_1 and X_2 ~$\mathcal{BD}$, X_3~$\mathcal{U}$ $(-1.5,15)$, X_4~$\mathcal{U}(-1,2)$, X_5~$\mathcal{U}(-15,1)$, X_6~$\mathcal{N}(2,0.2^2)$, X_7~$\mathcal{U}(-3,3)$, X_8~$\mathcal{U}(-15,1.5)$, X_9~$\mathcal{U}(-2,15)$, X_{10}~$\mathcal{U}(-2,15)$.

Table 20. Test functions.

(p = 3, 4, 5) for both methods. MCS with 10^8 runs is conducted to benchmark the accuracy of both methods. In Case 1, the probabilistic distributions of all the random input variables are unknown and only a number of raw data (10^5) exist, which cannot be solved by the traditional PCE methods, such as gPCE. Clearly, the more the raw data, the more reliable the results will be. Considering that the main objective of this paper is to investigate the effectiveness and capability of sDD-PCE in reducing the computational cost, it is assumed that a large number of raw data (10^5) exist of the random inputs.

The results are listed in **Tables 21–23**, in which e_m and e_v, respectively, denote the errors (%) of mean and variance relative to the results of MCS, N denotes the number of total sample points (function evaluations) used for PCE coefficients estimation during regression, and Na represents that the result cannot be obtained.

From the results some noteworthy observations are made. First, generally with high PCE order (p = 5), the results of sDD-PCE are accurate. Second, for low-dimensional problem (d = 2, Function 1), the efficiency and accuracy of sDD-PCE and fDD-PCE are almost comparable. Specially, for lower orders p = 3 and 4, sDD-PCE is even less efficient. The interpretation is that

	***f*DD-PCE**			***s*DD-PCE**		
e_m	**0.321**	0.099	0.044	0.330	0.201	0.181
e_v	0.232	0.813	0.173	0.203	0.099	0.068
p	3	4	5	3	4	5
N	**10**	15	21	20	30	20

Table 21. Results of function 1 (Case 1).

	***f*DD-PCE**			***s*DD-PCE**		
e_m	6.162	Na	Na	8.803	7.263	2.402
e_v	10.182	Na	Na	16.670	5.026	8.882
p	3	4	5	3	4	5
N	56	126	252	20	20	30

Table 22. Results of function 2 (Case 1).

	***f*DD-PCE**			***s*DD-PCE**		
e_m	Na	Na	Na	0.045	0.739	0.239
e_v	Na	Na	Na	18.134	12.882	2.479
p	3	4	5	3	4	5
N	286	1001	3003	30	30	30

Table 23. Results of function 3 (Case 1).

in addition to the regression process, the sample points are also required during the construction of the sparse PCE model for sDD-PCE, while for fDD-PCE, the sample points are only used during regression. Moreover, for em with $p = 3$ (lower order) of Function 1, fDD-PCE is even more accurate with higher efficiency (see underlined numbers). The reason may be that for low-dimensional problems with low-order PCE models, the size of the total polynomial terms is already small and the sparse structure of sDD-PCE is of little help in reducing the number of sample points since additional sample points are required during the selection of important polynomial terms. Therefore, fDD-PCE may produce more accurate results than sDD-PCE since it maintains more information. This will be verified later. Third, with the increase of dimension (from $d = 2, 5$ to $d = 10$), $\boldsymbol{N}$ is increased significantly with the increase of p for fDD-PCE, causing matrix ill-conditioned problem. So some results ($p = 4$ and 5) even cannot be obtained by fDD-PCE. Specially, for Function 3, the dimension is high ($d = 10$), fDD-PCE cannot produce results for any order p. However, for sDD-PCE, no remarkable increase in $\boldsymbol{N}$ is noticed since it adopts a sparse PCE model that can adaptively remove the insignificant polynomial terms, while its accuracy is generally improved clearly exhibiting small error relative to MCS. When $p = 5$, only 13 polynomial terms are selected from 3003 total terms for Function 3; while for Function 1, 4 are selected from 21 total terms. Therefore, the larger the dimension, the more obvious the advantage of sDD-PCE over fDD-PCE in efficiency.

In Case 2, the PDFs of all the random inputs are known and assumed to follow common distributions. This is a general case that can be solved by the traditional probabilistic distribution-based PCE methods. The results are shown in **Tables 24–26**. Generally with high PCE order ($p = 5$), the results of sDD-PCE are accurate, demonstrating its effectiveness in dealing with random inputs with known PDFs. Meanwhile, for low-dimensional problem (Function 1), generally sDD-PCE is more accurate with the similar N as fDD-PCE. However, for lower order ($p = 2$) of Function 1, fDD-PCE is even more accurate than sDD-PCE, but with

	***f*DD-PCE**			***s*DD-PCE**		
e_m	**0.083**	0.044	0.060	0.710	0.010	0.100
e_v	**0.468**	0.758	0.211	0.975	0.061	0.061
p	3	4	5	3	4	5
N	**10**	15	21	30	15	20

Table 24. Results of function 1 (Case 2).

	***f*DD-PCE**			***s*DD-PCE**		
e_m	24.401	Na	Na	1.244	0.490	0.216
e_v	39.578	Na	Na	4.380	3.271	2.837
p	3	4	5	3	4	5
N	56	126	252	20	20	30

Table 25. Results of function 2 (Case 2).

	fDD-PCE			sDD-PCE		
e_m	Na	Na	Na	3.461	4.432	0.317
e_v	Na	Na	Na	20.155	6.217	4.223
p	3	4	5	3	4	5
N	286	1001	3003	30	30	30

Table 26. Results of function 3 (Case 2).

much smaller N. This observation is consistent with what has been noticed in Case 1 and the reason is that additional sample points are required to selecting important polynomial terms. With the increase of dimension, N is increased significantly with the increase of p for fDD-PCE. However, for sDD-PCE, remarkable improvement in the accuracy is noticed without a remarkable increase in N. These results show great agreements to what has been noticed in Case 1.

In Case 3, the PDFs of all the random inputs are known; however, some of them follow nontrivial distributions. In this case, the traditional gPCE method cannot work well since large errors would be induced in transforming such nontrivial distributions to certain ones in the Askey scheme. The results are shown in **Tables 27–29**, which exhibit great agreements to what has been observed in Case 1 and Case 2. The proposed sDD-PCE method can significantly reduce the number of sample points while with high accuracy. The higher the dimension, the more advantageous the adaptive sparse structure of sDD-PCE can be. In this case, only 11 polynomial terms are selected from 3003 total terms for $d = 10$ with sDD-PCE. Moreover, sDD-PCE can produce accurate and efficient results for nontrivial distributed random inputs.

To verify the guess that for low-dimensional problems with low-order PCE models, fDD-PCE may produce more accurate results than sDD-PCE since it maintains more information.

	fDD-PCE			sDD-PCE		
e_m	1.210	0.854	0.302	1.366	1.044	0.161
e_v	2.321	0.748	0.815	0.805	0.161	0.000
p	3	4	5	3	4	5
N	10	15	21	10	10	10

Table 27. Results of function 1 (Case 3).

	fDD-PCE			sDD-PCE		
e_m	3.324	Na	Na	5.718	1.383	0.680
e_v	7.855	Na	Na	7.634	7.322	2.290
p	3	4	5	3	4	5
N	56	126	252	20	30	30

Table 28. Results of function 2 (Case 3).

	fDD-PCE			sDD-PCE		
e_m	Na	Na	Na	4.114	2.212	0.112
e_v	Na	Na	Na	48.894	15.817	3.101
p	3	4	5	3	4	5
N	286	1001	3003	30	30	30

Table 29. Results of function 3 (Case 3).

Another test is conducted for Function 1 with lower order $p = 2$ with all the three cases, of which the results are shown in **Table 30**. Just as expected, fDD-PCE is clearly more accurate than sDD-PCE while generally with less sample points. For Function 1 with $p = 2$, the total number of polynomial terms is 6, which is very small. With sDD-PCE, only the last polynomial term is removed, while more points are required in removing the insignificant polynomials. So the sparse scheme does not have obvious impact under this circumstance. Therefore, it is concluded that the developed sDD-PCE method is particularly applicable to high-dimensional problems, especially those requiring a high order PCE model.

	Case 1		Case 2		Case 3	
	fDD	sDD	fDD	sDD	fDD	sDD
e_m	0.2801	0.146	0.0366	0.244	0.414	0.807
e_v	0.6344	0.367	0.3577	0.431	0.552	0.477
N	6	7	6	10	6	18

Table 30. Results of function 1 ($p = 2$).

3.3. Summary

The developed sDD-PCE can reduce the number of polynomial terms in the PCE model, thus reducing the computational cost. Generally, the larger the random input dimension, the more obvious the advantage of the developed sDD-PCE over fDD-PCE in efficiency. The sDD-PCE method is much more efficient than fDD-PCE in solving high-dimensional problems, especially those requiring a high order PCE model.

4. Sparse DD-PCE-based robust optimization using trust region

In Section 3, to reduce the computational cost of DD-PCE, a sparse DD-PCE method has been developed by removing some insignificant polynomial terms from the full PCE model, thus decreasing the number of samples for regression in computing PCE coefficients. However, when the sparse DD-PCE is applied to robust optimization, it is conventionally a triple-loop process (see **Figure 9**): the inner one tries to identify the insignificant polynomial terms of the

http://dx.doi.org/10.5772/intechopen.68484

PCE model (the dash box); the middle is UP; the outer is the search for optima, which clearly is still very time-consuming for problems with expensive simulation models.

As has been mentioned in Section 3, during each optimization iteration, although the sample points required for regression during UP of sDD-PCE are greatly reduced, certain additional number of sample points are required to identify the insignificant polynomial terms by the inner loop. If at some iteration design points, almost the same sparse polynomial terms are retained, the inner loop can clearly be avoided, thus saving the computational cost. To address this issue, the trust region technique widely used in nonlinear optimization is extended in this section. During optimizing, a trust region is dynamically defined. If the updated design point lies in the current trust region, it is considered that the insignificant terms of its PCE model remain unchanged compared to those of the last design point, i.e., the inner loop is eliminated at the updated design point. Meanwhile, to further save the computational cost, the sample points lying in the overlapping area of two adjacent sampling regions are reused for the PCE coefficient regression for the updated design point. The proposed robust optimization procedure employing sparse DD-PCE in conjunction with the trust region scenario is applied to several examples of robust optimization, of which the results are compared to those obtained by the robust optimization without the trust region method, to demonstrate its effectiveness and advantage.

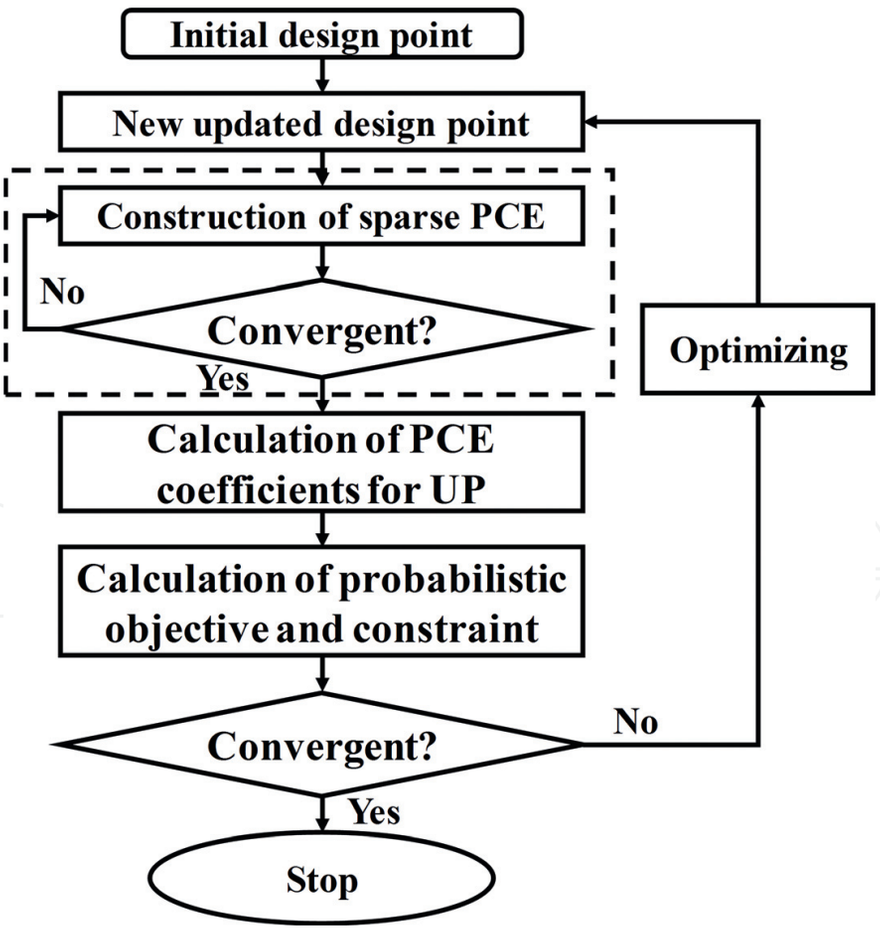

Figure 9. The triple-loop formulation of sDD-PCE-based robust optimization.

4.1. The trust region scenario

The trust region method is a traditional approach that has been widely used in nonlinear numerical optimization [28]. The basic idea of the trust region method is that in the trust region of the current iteration design point, the second-order Taylor expansion is used to approximate the original objective function. If the accuracy of the current second-order Taylor expansion is satisfied, the size of the trust region is increased to speed up the convergence, and if not it is reduced to improve the accuracy of approximation. To reduce the computational cost of design optimization, the idea of the trust region technique has been extended and applied to reliability-based wing design optimization [29], multifidelity wing aero-structural optimization [30], and multifidelity surrogate-based wing optimization [31], which has been widely believed as an efficient strategy in design optimization. For example, when the trust region technique is applied to meta-model-based design optimization, during optimization, the sample points are sequentially generated in the trust region and the radius of the trust region is dynamically adjusted based on the accuracy of the meta-model in the local region.

4.2. Robust design using sparse data-driven PCE and trust region

The scenario of trust region is extended here to reduce the computational cost of sDD-PCE-based robust optimization. The basic idea is that the radius of a trust region is determined by the distance between two successive design points and the variation of the corresponding objective function values. If the updated design point $\mu_{\mathbf{x}}^{k+1}$ lies in the current trust region, it is considered that the insignificant terms of its PCE model remain unchanged compared to those of the last design point $\mu_{\mathbf{x}}^{k}$, i.e., the inner loop is eliminated at the updated design point. Meanwhile, the sample points lying in the overlapping area of two adjacent sampling regions are reused for the PCE coefficient regression for the updated design point to further save the computational cost. Generally, for a practical engineering optimization problem, there is only one performance function that is computationally expensive. Therefore, only one PCE model is required to be constructed and the UP for the rest of the functions can be conveniently implemented by MCS. In this study, it is assumed that the PCE model is only constructed for the objective function and the general steps of the proposed method is as below.

Step 0: Set the iteration number as $k = 1$ and the initial staring design point $\mu_{\mathbf{x}}^{0}$, do robust optimization with sDD-PCE without trust region and obtain a new design variable $\mu_{\mathbf{x}}^{k}$, where the Latin Hypercube sample points are generated around $\mu_{\mathbf{x}}^{0}$ to calculate the PCE coefficients.

Step 1: After the kth optimization iteration, define/update the trust region at the current obtained new design point $\mu_{\mathbf{x}}^{k}$ as a rectangle with each length as

$$r_1 = max\left\{\zeta_1\left|\mu_{\mathbf{x}}^{k}\right|_2, \zeta_2\right\}, r_2 = max\left\{\zeta_1\left|Y^k\right|, \zeta_2\right\} \tag{24}$$

where $|\mu_{\mathbf{x}}^{k}|_2 = \sqrt{\sum_{i=1}^{d}(\mu_{\mathbf{x}_i}^{k})^2}$ and $|Y^k|$ is the absolute value of corresponding objective function at $\mu_{\mathbf{x}}^{k}$, i.e., $|Y^k| = abs\left(Y(\mu_{\mathbf{x}}^{k})\right)$; ζ_1 and ζ_2 are user-defined parameters, which can be constants or functions with respect to the iteration number k.

Step 2: During the $(k + 1)$th optimization iteration, the obtained new design point is $\mu_{\mathbf{x}}^{k+1}$. Before conducting UP, calculate the variation between two successive design points $\mu_{\mathbf{x}}^{k}$ and $\mu_{\mathbf{x}}^{k+1}$ as $\Delta\mathbf{x} = |\mu_{\mathbf{x}}^{k+1} - \mu_{\mathbf{x}}^{k}|_2 = \sqrt{\sum_{i=1}^{d}(\mu_{\mathbf{x}_i}^{k+1} - \mu_{\mathbf{x}_i}^{k})^2}$ and the variation of the objective function $\Delta Y = |Y^{k+1}(\mu_{\mathbf{x}}^{k+1}) - Y^{k}(\mu_{\mathbf{x}}^{k})|$.

Step 3: If $\Delta\mathbf{X} \le r_2$ and $\Delta Y \le r_2$ both are satisfied, $\mu_{\mathbf{x}}^{k+1}$ is considered to be located in the trust region of $\mu_{\mathbf{x}}^{k}$ defined in Eq. (45), and go to Step 4; if either $\Delta\mathbf{X} \le r_1$ or $\Delta Y \le r_2$ cannot be satisfied, $\mu_{\mathbf{x}}^{k+1}$ is considered to be not located in the trust region of $\mu_{\mathbf{x}}^{k}$ defined in Eq. (45), and go to Step 5.

Step 4: The retained polynomial terms $\Phi_i(\boldsymbol{x})$ at the updated new design point $\mu_{\mathbf{x}}^{k+1}$ are kept as the same as those for the last design point $\mu_{\mathbf{x}}^{k}$, indicating that the inner loop of UP conducted on $\mu_{\mathbf{x}}^{k+1}$ is removed. The Latin Hypercube sample points are generated around $\mu_{\mathbf{x}}^{k+1}$ according to the distribution type and parameters of $\mathbf{X}$ with the same number of sample points as that used at the last design point $\mu_{\mathbf{x}}^{k}$ to calculate the PCE coefficients. Meanwhile, the sample points located in the overlapping area of the two successive sampling regions are identified and reused for PCE coefficients calculation to improve the accuracy.

Step 5: The inner loop is conducted on the updated design point $\mu_{\mathbf{x}}^{k+1}$ to detect the significant polynomial terms. Similarly, the sample points located in the overlapping area of the two successive sampling regions are also reused at the updated design point $\mu_{\mathbf{x}}^{k+1}$ in detecting the significant polynomial terms and calculating the PCE coefficients to save the computational cost.

Step 6: Set $k = k + 1$, based on the results of UP, search for the next updated new design point $\mu_{\mathbf{x}}^{k+1}$ and go to Step 1.

The above procedure will continue until the convergent criterion is satisfied. **Figure 10** shows the case that the sample points in the previous optimization iteration are reused in the two successive iterations. As is seen that two points are located in the overlapping area of two successive sampling regions, thus are reused in the next iteration for regression to identify the significant polynomials/calculate the PCE coefficients. In this way, the computational cost can be further reduced.

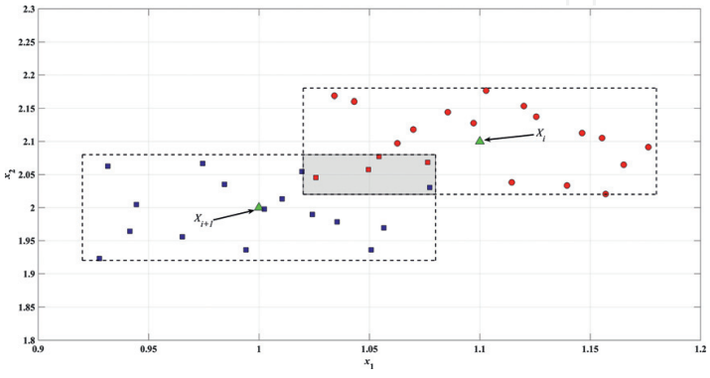

Figure 10. Illustration of the reuse of sample points.

4.3. Comparative studies

The first example is the Ackley Function:

$$f(\mathbf{X}) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{j=1}^{d}X_j^2}\right) - \exp\left(\frac{1}{d}\sum_{j=1}^{d}\cos(2\pi X_j)\right) + 22.71282, d = 10 \tag{25}$$

The robust design optimization of this example is:

$$\begin{aligned} &\min F = \mu_f + k\sigma_f \\ &-10 \le \mu_{X_j} \le 10, j = 1, 2, \ldots, 10 \end{aligned} \tag{26}$$

All the design variables are considered to follow uniform distribution with variation of ±0.2 around their mean values and k in Eq. (26) is set as k = 20. In this study, the fmincon function in Matlab is used for optimization, and ζ_1 and ζ_2 in Eq. (45) are set as ζ_1 = 0.5 and ζ_2 = 0.5. Meanwhile, the obtained optimal design variables of sDD-PCE-based robust design with and without trust region scenario as well as the deterministic design without considering any uncertainties are respectively substituted into Eq. (26) through MCS (with 1e^6 runs) to calculate the mean μ_f and standard deviation σ_f of the objective function.

The results are shown in **Table 31**, from which it is found that compared to the robust optimization without the trust region scenario (denoted by without), the obtained performance results (μ_f, σ_f, and F) of the robust optimization with the trust region scenario (denoted by with) are comparable. However, the number of function calls (denoted as Funcall) is clearly reduced. The decrease in computational cost is attributed to the application of trust region scenario and the reusing of sample points. Meanwhile, the optimal designs of the two robust designs are both less sensitive to uncertainties (smaller σ_f) compared to the results of deterministic design (denoted by DD). These results demonstrate the effective and advantage of the proposed method.

The second example is the robust design optimization of an automobile torque arm, shown in **Figure 11**.

In this problem, the four geometrical parameters (a, d_1, d_2, and l) are considered as design variables, and the yielding strength S_y, Young's modulus E, and the applied force Q are deterministic parameters.

$$\begin{aligned} &\min f(a, d_1, d_2, l) = \frac{\pi a d_2^2}{4} + 2\left(l - \frac{d_1}{2} - \frac{d_2}{2}\right)a^2 \\ &s.t.\ g_1(a, d_2, l) = \frac{Q(2l - d_2)d_2}{4IS_y} - 1.0 \le 0 \\ &\quad g_2(a, d_1, d_2, l) = 1.0 - \frac{\pi^2 E a^4}{3(2l - d_1 - d_2)^2}\frac{d_2 - d_1}{Ql} \le 0 \\ &\quad 5 \le a \le 15, 45 \le d_1 \le 55, 55 \le d_2 \le 65, 110 \le l \le 210 \end{aligned} \tag{27}$$

	μ_X^*	μ_f	σ_f	F	**Funcall**
DD	[0,0,0,0,0,0,0,0,0,0]	1.8839	0.4390	10.6639	—
with	[0.6246,0.7066,0.6687,0.7796,0.5744, 0.6784,0.7470,0.6333,0.6578,0.6904]	4.5014	0.1377	7.2554	12,735
without	[0.6564,0.6935,0.6984,0.7036,0.6691, 0.0299,0.0141,0.6407,0.0205,0.0038]	3.7457	0.2003	7.7517	16,840

Table 31. Results of the Ackley Function.

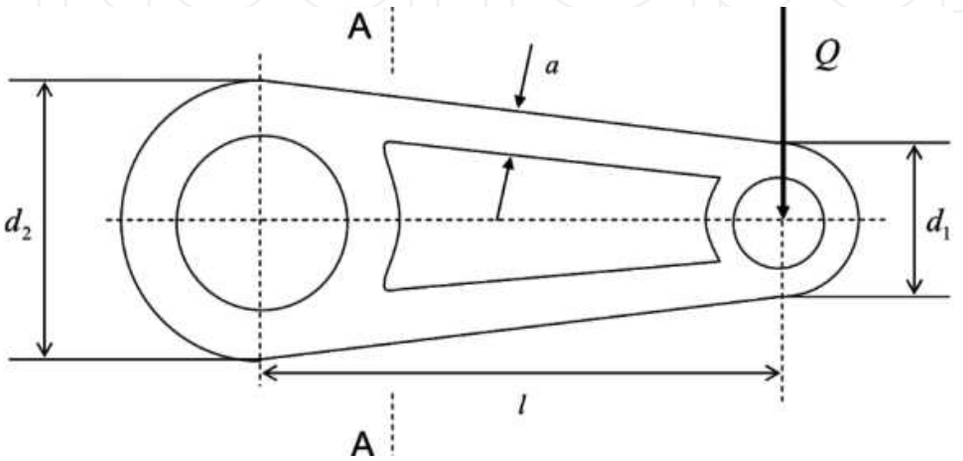

Figure 11. Automobile torque arm.

where the objective function f represents the volume of the arm, the first constraint g_1 denotes the yielding failure at section A-A, the second constraint g_2 denotes the buckling failure at the two connecting rods, and $I = a^2(d_2 - a)^2/2 + a^4/6$.

The distribution parameters of the four design variables and design parameters are shown in **Table 32**.

Random variables	**Distribution**	**Lower bound**	**Upper bound**
a	Uniform	μ_a −0.5 mm	μ_a +0.5 mm
d_1	Uniform	μ_{d1} −0.5 mm	μ_{d1} +0.5 mm
d_2	Uniform	μ_{d2} −0.5 mm	μ_{d2} +0.5 mm
l	Uniform	μ_l −0.5 mm	μ_l +0.5 mm
Parameters		**Values**	
Q	Deterministic	5500 N	
S_y	Deterministic	170 N/mm^2	
E	Deterministic	2.1×10^{10} N/mm^2	

Table 32. Distribution parameters for design variables and design parameters.

The corresponding robust design optimization model is formulated as

$$\begin{aligned} &\min F = \omega_1 \frac{\mu_f}{\mu_f^*} + \omega_2 \frac{\sigma_f}{\sigma_f^*}, \omega_1 = 0.5, \omega_2 = 0.5 \\ &s.t. \quad G_1(a, d_2, l) = \mu_{g_1} + k\sigma_{g_1} \le 0 \\ &\qquad G_2(a, d_1, d_2, l) = \mu_{g_2} + k\sigma_{g_2} \le 0 \\ &\qquad 5 \le \mu_a \le 15, 45 \le \mu_{d_1} \le 55, 55 \le \mu_{d_2} \le 65, 110 \le \mu_l \le 210 \end{aligned} \tag{28}$$

As has been mentioned above, the PCE model is only constructed for the objective function and the results are shown in **Table 33**. It is noticed that the robust optimization designs with and without the trust region scenario yields comparable results, while the function calls (objective function calls) required by design with trust region is evidently smaller. The deterministic design cannot even obtain a feasible optimal solution with both constraint violated (>0), since it does not consider uncertainties during design. These results further demonstrate the effectiveness and advantage of the proposed method.

4.4. Summary

The employment of the trust region in sDD-PCE-based robust optimization can evidently reduce the computational cost. However, the determination of the trust region in this chapter is still very subjective and a more rigorous method should be explored. In this section as well as Section 3, the scenarios of sparse PCE and trust region are only employed to DD-PCE to save the computational cost. However, the methods proposed here are also applicable to other PCE approaches, such as gPCE and GS-PCE.

In this chapter, the latest advances in PCE theory and approach for probabilistic UP are comprehensively presented in detail. However, it does not limit the application of PCE to nonprobabilistic UP to address epistemic uncertainties. Sudret and Schöbi have proposed a two-level metamodeling approach using nonintrusive sparse PCE to surrogate the exact computational model to facilitate the uncertainty quantification analysis, in which the input variables are modeled by probability-boxes (*p*-boxes), accounting for both aleatory and epistemic uncertainty [32]. The Fuzzy uncertainty propagation in composites has been implemented using Gram-Schmidt polynomial chaos expansion, in which the parameter uncertainties are represented by fuzzy membership functions [5]. A general framework has been proposed for a dynamical uncertain system to deal with both aleatory and epistemic uncertainty using PCE, where the uncertain parameters are described through random variables and/or fuzzy variables [33]. The mix UP approach is proposed, in which the inner loop PDFs are calculated using the PCE, and outer loop bounds can be computed with optimization-based interval

	μ_X^*	μ_f	σ_f	F	G_1	G_2	Funcall
DD	[8.13, 55.00, 55.00, 110.00]	2.6616e^4	1.2171e^3	1	**0.1848**	**5.1509e^4**	82
with	[8.53, 54.10, 58.67, 111.03]	3.1027e^4	1.3355e^3	1.1315	−0.0123	−1.1833e^5	658
without	[8.57, 52.68, 57.50, 110.00]	3.0332e^4	1.3093e^3	1.1077	−4.0000e^{-4}	−1.2913e^2	1283

Table 33. Results of automobile torque arm.

estimation [34]. PCE has also been applied for solving Bayesian inverse problem as "surrogate posterior." However, it has been indicated that the accuracy cannot always be ensured by PCE since a sufficiently accurate PCE for this problem requires a high order, making PCE impractical compared to directly sampling the posterior [35].

Author details

Shuxing Yang*, Fenfen Xiong and Fenggang Wang

*Address all correspondence to: yangshx@bit.edu.cn

School of Aerospace Engineering, Beijing Institute of Technology, Beijing, China

References

[1] Matthies HG. Quantifying uncertainty: Modern computational representation of probability and applications. Extreme Man-Made and Natural Hazards in Dynamics of Structures. Springer Netherlands, 2007;105–135

[2] Kiureghian AD, Ditlevsen O. Aleatory or epistemic? Does it matter? Structural Safety. 2009;**31**(2):105–112

[3] Swiler LP, Romero VJ. A survey of advanced probabilistic uncertainty propagation and sensitivity analysis methods. Proposed for presentation at the 2012 Joint Army Navy NASA Air Force Combustion/Propulsion Joint Subcommittee Meeting; December 3-7, 2012; Monterey, CA

[4] Du X, Chen W. A most probable point-based method for efficient uncertainty analysis. Journal of Design & Manufacturing Automation. 2001;**4**(1):47–66

[5] Mukhopadhyay S, Khodaparast H, Adhikari S. Fuzzy uncertainty propagation in composites using Gram–Schmidt polynomial chaos expansion. Applied Mathematical Modelling. 2016; **40**(7–8):4412–4428

[6] Jiang C, Zheng J, Ni BY, Han X. A probabilistic and interval hybrid reliability analysis method for structures with correlated uncertain parameters. International Journal of Computational Methods. 2015;**12**(4):1540006 (24 pages)

[7] Terejanu G, Singla P, Singh T, Scott PD. Approximate interval method for epistemic uncertainty propagation using polynomial chaos and evidence theory. IEEE American Control Conference; 30 June–2 July 2010; Marriott Waterfront, Baltimore, MD, USA.

[8] Lee SH, Chen W. A comparative study of uncertainty propagation methods for black-box-type problems. Structural & Multidisciplinary Optimization. 2009;**37**(3):239–253

[9] Dodson M, Parks GT. Robust aerodynamic design optimization using polynomial chaos. Journal of Aircraft. 2009;**46**(2):635–646

[10] Coelho R, Bouillard P. Multi-objective reliability-based optimization with stochastic metamodels. Evolutionary Computation. 2011;**19**(4):525–560

[11] Xiu D, Karniadakis GE. The wiener-askey polynomial chaos for stochastic differential equations. SIAM Journal on Scientific Computing. 2002;**24**(2):619–644

[12] Wiener N. The homogeneous chaos. American Journal of Mathematics. 1938;**60**(1):897–936

[13] Fan et al. Parameter uncertainty and temporal dynamics of sensitivity for hydrologic models: A hybrid sequential data assimilation and probabilistic collocation method. Environmental Modelling & Software. 2016;**86**:30–49

[14] Guerine A, Hami AE, Walha L, et al. A polynomial chaos method for the analysis of the dynamic behavior of uncertain gear friction system. European Journal of Mechanics - A/Solids. 2016;**59**:76-84

[15] Meecham WC, Siegel A. Wiener-Hermite expansion in model turbulence at large Reynolds numbers. Physics of Fluids (1958-1988). 1964;**7**(8):1178–1190. DOI: 10.1063/1.1711359

[16] Witteveen JAS, Bijl H. Modeling arbitrary uncertainties using Gram-Schmidt polynomial chaos. 44th AIAA Aerospace Sciences Meeting and Exhibit; 9–12 January 2006; Reno, Nevada

[17] Wan X, Karniadakis GE. Multi-element generalized polynomial chaos for arbitrary probability measures. SIAM Journal on Scientific Computing. 2006;**28**(3):901–928

[18] Oladyshkin S, Nowak W. Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion. Reliability Engineering & System Safety. 2012;**106**(4):179–190

[19] Hosder S, Walters RW, Balch M. Efficient sampling for non-intrusive polynomial chaos applications with multiple uncertain input variables. 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference; 23–26 April 2007; Honolulu, Hawall

[20] Eldred MS. Recent advances in non-intrusive polynomial chaos and stochastic collocation methods for uncertainty analysis and design. 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference; 4–7 May 2009; Palm Springs, California

[21] Abramowitz M, Stegun I, Mcquarrie D A. Handbook of Mathematical Functions. Dover Publications, New York,1964

[22] Xiong F, Greene S, Chen W, Xiong Y, Yang S A new sparse grid based method for uncertainty propagation. Structural & Multidisciplinary Optimization. 2009;**41**(3):335–349

[23] Efron B, Hastie T, Johnstone I, Tibshirani R. Least angle regression. Mathematics. 2004;**32**(2):407–499

[24] Tatang MA, Pan W, Prinn RG, McRae GJ. An efficient method for parametric uncertainty analysis of numerical geophysical models. Journal of Geophysics Research. 1997;**102**(D18):21925–21932

[25] Hu C, Youn BD. Adaptive-sparse polynomial chaos expansion for reliability analysis and design of complex engineering systems. Structural & Multidisciplinary Optimization. 2011; **43**(3):419–442

[26] Wan X, Karniadakis GE. An adaptive multi-element generalized polynomial chaos method for stochastic differential equations. Journal of Computational Physics. 2005;**209** (2):617–642

[27] Tu J, Cheng YP. An integrated stochastic design framework using cross-validated multivariate metamodeling methods. SAE Technical Paper 2003-01-0876; 2003

[28] Nocedal J, Wright S. Numerical Optimization. Springer Series in Operations Research and Financial Engineering. New York: Springer; 2006

[29] Elham A, Tooren MJLV. Trust region filter-SQP method for multi-fidelity wing aerostructural optimization. Variational Analysis and Aerospace Engineering. 2016;**116**:247–267

[30] Kim S, Ahn J, Kwon JH. Reliability based wing design optimization using trust region framework. 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference; 30 August–1 September 2004; Albany, New York

[31] Robinson TD, Eldred MS, Willcox KE, Haimes R. Surrogate-based optimization using multifidelity models with variable parameterization and corrected space mapping. AIAA Journal. 2008;**46**(11):2814–2822

[32] Schöbi R, Sudret B. Uncertainty propagation of p-boxes using sparse polynomial chaos expansions. 2016, **339**:307–327

[33] Jacquelin E, Friswell MI, Adhikari S, Dessombz O, Sinou J. Polynomial chaos expansion with random and fuzzy variables. Mechanical Systems and Signal Processing. 2016;**75** (15):41–56

[34] Eldred MS, Swiler LP, Tang G. Mixed aleatory-epistemic uncertainty quantification with stochastic expansions and optimization-based interval estimation. Reliability Engineering and System Safety. 2011;**96**(9):1092–1113

[35] Lu F, Morzfeld M, Tu X, Chorin AJ. Limitations of polynomial chaos expansions in the Bayesian solution of inverse problems. Journal of Computational Physics. 2014;**282** (C):138–147

IntechOpen